PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G07F 7/00

A2

(11) International Publication Number: **WO 98/58356**

(43) International Publication Date: 23 December 1998 (23.12.98)

(21) International Application Number: PCT/US98/12408

(22) International Filing Date: 16 June 1998 (16.06.98)

(30) Priority Data:
60/049,783 16 June 1997 (16.06.97) US

(71)(72) Applicant and Inventor: KEILANI, Badieh, Z., II [US/US]; 160 Central Park South, New York, NY 10019 (US).

(74) Agents: GARRETT, Arthur, S. et al.; Finnegan, Henderson, Farabow Garrett & Dunner, L.L.P., 1300 I Street, N.W., Washington, DC 20005-3315 (US).

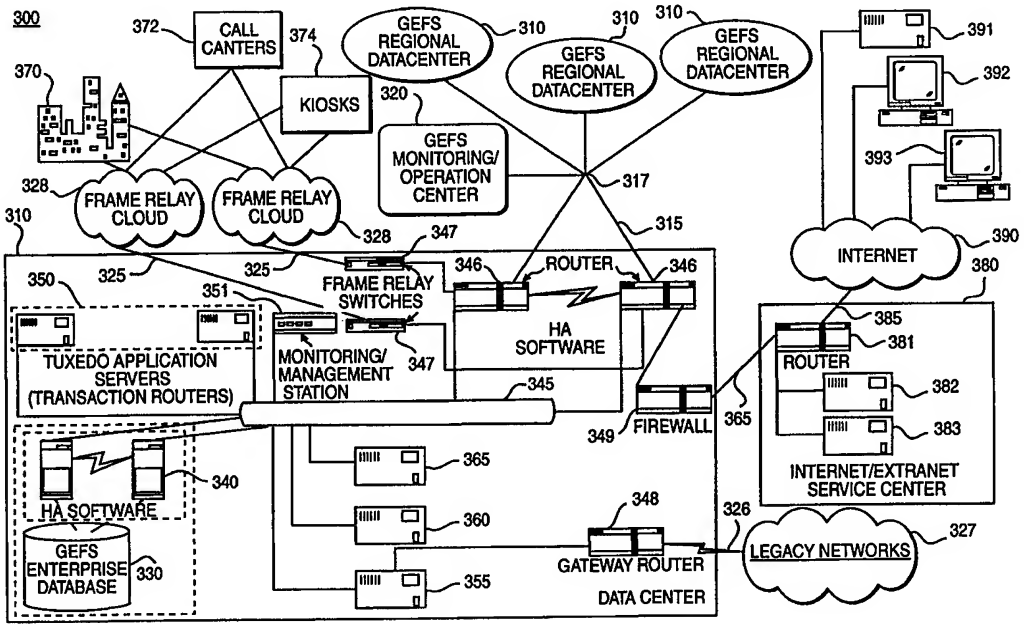
(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, GW, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: SYSTEM AND METHOD FOR PROCESSING MULTIPLE FINANCIAL APPLICATIONS USING A THREE-TIER VALUE NETWORK

(57) Abstract

A three-tier, client-server system and method provides improved financial services over a network and processes financial transactions among many user accounts and service providers, and automatically routes and receives financial transaction messages or other financial or account information, reviews account information and other financial data, and updates the financial records of user accounts and service providers in accordance with the particular financial transaction being processed. The system and method also allows access and analysis of financial data to provide improved financial services over a computerized network system. The financial transaction network includes a data center operations portion containing business critical computing module, a back office, general accounting and smart suite financial applications module; and a data warehouse and decision support module. The data center operations can be accessed by, and communicates through, an enterprise network backbone. The system also includes middle-tier application servers which are also accessed by, and communicate with the enterprise network backbone. Application servers can further be accessed by, and communicate with the bottom tier, front end clients over a proprietary virtual private extranet. And application servers can also further be accessed by the Internet, through the World Wide Web, and Java applets.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

SYSTEM AND METHOD FOR PROCESSING MULTIPLE FINANCIAL APPLICATIONS USING A THREE-TIER VALUE NETWORK

I. BACKGROUND OF THE INVENTION

A. Field of the Invention

This invention relates generally to processing financial transactions and financial information over a network. More particularly, the invention relates to a method and apparatus for processing multiple financial transactions and multiple financial information over a three-tier value network.

B. Description of the Related Art

In today's global marketplace, the financial services industry utilizes various financial applications to process myriad financial transactions and financial information. The financial services industry includes banks, thrifts, credit unions, financial services providers, securities firms, insurance companies, brokerage houses, and other financial institutions. The financial services industry has traditionally been divided into three distinct segments: banking, securities brokerage, and insurance. Financial applications include any application of software or hardware to prepare, process, implement, or transmit financial transaction or financial information. Financial transactions include business transactions—an exchange of information or value—such as account balances, interest calculations, and currency exchange rates, and other related information. Financial information includes information systems transactions—an atomic unit of work—such as a transformation/calculation of a piece of data and its subsequent update to a database.

Typically, a member of the financial services industry (i.e., a financial services provider) establishes an internal system for processing financial applications and other financial data. These stand-alone applications are generally specific to a particular internal system and include proprietary data storage capabilities. As a result, one of these stand-alone financial applications generally cannot access or utilize another application or even another internal system. The need to manage and analyze the data from different financial applications has become essential. Nevertheless, there is

still a "culture of separateness" in the financial services industry. Indeed, no single financial services provider within the financial services industry fully comprehends or exploits current technology. Each provider wants to invent their own distribution system and monopolize it. As a result, many problems have arisen, such as, the Year 2000 (Y2K) problem. Most financial institutions cannot commit the necessary resources to solving the Y2K problem, yet federal regulators have informed financial institutions that they will be held responsible for Y2K compliance.

In addition, the pace and number of financial transactions and financial information used by financial services providers has grown at a staggering rate. Although advanced data processing (ADP) and networks have increased the efficiency and accuracy by which providers handle financial transactions, neither the efficiency nor accuracy is as great as it needs to be. Additionally, many businesses want better interaction with service bureaus, such as, for example, ADP. In addition, financial services providers wish to provide branded credit cards, debit cards, and smart cards.

Furthermore, the demand for financial information has likewise experienced incredible changes. Financial services providers have traditionally enjoyed a profitable existence without pressure to adopt new technologies. However, customers are demanding ever more convenience. Also, regulatory barriers separating the industry elements are disappearing, and non-traditional competitors are redefining service and price. Industry deregulation, the maturation of a technology-savvy generation of consumers, and the increasing availability of powerful information networks and technologies—like the Internet—have combined to create tremendous potential for delivering a wide spectrum of non-traditional financial services offerings to consumers (e.g., mutual funds sales by banks, loans by insurance companies and deposit services by brokerage houses).

Consumers are also demanding more and different financial services. Indeed, consumers want access to their financial information and the capability to process financial transactions at their own convenience, without having to go to a bank or other financial institution. Consumers also want to use telephones, automated teller machines (ATMs), and PC-based or Internet banking, and want access to a wide range of Internet financial services, including mortgage lenders, securities providers, and mutual funds. To address these demands and this changing environment, financial services providers must protect the existing customer base and provide a predominant delivery system for a comfortable and accepted feature of the financial services infrastructure.

In the past few years, however, the financial services industry has faced great changes in the traditional methodologies of conducting business. At the same time, advances in information technology are enabling even more financial services delivery models. For example, customers now have PCS in the home, have access to software tools such as Intuit Quicken® and Microsoft Money®, and also utilize the Internet and the World Wide Web. In addition, many consumers now utilize call centers to access financial transactions and financial information.

Unfortunately, for many financial services providers, two factors—investment in proprietary technologies and/or lack of ongoing investment in technology and process change—have paralyzed such companies, enforcing a rigid business posture that prevents them from capitalizing on environmental changes, or at least from doing so cost-effectively. Consequently, many such companies are experiencing erosion of their customer base as consumers migrate to those few financial services companies that offer cheaper, more convenient, and more comprehensive services. Without vast capital resources necessary to change their underlying computing and business

infrastructure, the majority of financial services providers face a virtually insurmountable barrier to processing financial information.

Figure 1 is a schematic diagram showing traditional banks providing financial services to consumers. As shown in Fig. 1, the applications used by banks to provide financial services can include a back office system, various teller stations, other work stations, such as IBM 3270 or PCS, and other customer interaction work stations, such as loan officers and administrators. Notably, the banks provide customers with financial services by using outside service providers. These service providers typically include a legacy mainframe or local area network (LAN) system and provide daily reports and output to their customers by printing them out, or transmitting them by electronic means, and also manage and handle external branch ATMs and other external interfaces links, such as Fedwire, Chips, SWIFT, and other automated clearinghouse (ACH) services. The banks can also use a standalone system connected to, typically, IBM 3270 or PC workstations and teller stations. The standalone systems interface with external information links.

However, banks and existing financial services providers are facing problems as they look ahead to the future. For example, the largest banks, or tier 1 banks, are burdened with ever increasing regulatory restrictions and have difficulty adding new financial services to their existing mainframe or LAN legacy systems. Tier 1 banks have assets in excess of $1 billion. Smaller tier 2 and 3 banks lack the capital resources to develop their own systems for providing new, high-tech financial services. Tier 2 banks have assets of $300 million to $1 billion, and tier 3 banks have assets up to $300 million. Tier 2 and 3 banks together comprise 95% of the total number of banks in the United States, yet collectively control only 23% of the asset base held by banks. Due to their

smaller size, tier 2 banks and tier 3 banks typically outsource the new financial services to existing financial services providers.

However, existing financial services providers have difficulty adding new financial services to their existing mainframe or LAN legacy systems. As a result, many computerized network systems process financial transactions and financial information for banks, thrifts, credit unions, securities firms, financial services firms, brokerages, and the like. However, these systems, typically either mainframe file servers or client-server LANs, cannot run several financial applications on the same network, and are limited in efficiently managing resources and consolidating data.

Also, financial services providers have traditionally charged exorbitant service fees that continue to increase the costs for banks to service their customers. One of the factors driving these high fees is the cost of developing and maintaining legacy management information systems that support the various lines of business. These legacy environments include older applications and inflexible direct-to-direct Wide Area Networks (WANs) that are expensive to maintain and enhance.

In addition, financial services providers are inadequately equipped to address the needs of many consumers, due to the recent changes in technology. Indeed, as the customers of financial services providers request more convenient and extensive access to their financial information at lower costs, banks feel increasingly competitive pressures to deliver financial information at the customers' convenience in order to avoid losing their customers to non-bank competitors. Many non-traditional financial organizations have or will soon provide financial services.

Furthermore, as financial services providers enhance, modify, or replace their internal bank systems, for example, to address Y2K problems, a need exists for a system that can provide flexibility, reliability, efficiency, and accuracy. To take the Y2K problem as an example, this is a

computing crisis of possible catastrophic proportions. Many mission-critical systems worldwide were originally coded with 2-digit year fields. The current programs have no concept of the difference between 1900 and 2000. As a consequence, business transactions that are date dependent will become meaningless when processed by such programs. It is not inconceivable that companies that do not address this problem through analysis and reprogramming will be unable to conduct business as of January 1, 2000 or sooner, possibly rendering them insolvent, and potentially breaking the infrastructure of the global information society. Therefore, Y2K presents a financially monumental opportunity for those organizations that can assist others in negating the problem.

Some attempts have been made to address these problems among the financial services providers in the financial services industry. Yet, with the present systems, it is not technologically possible to create virtual financial service networks—linking financial information located without respect to geography, and delivering that information to the point of customer contact, wherever the point of contact. Still, if financial institutions—banks, brokerage houses, and insurance companies—were able to access all of a customer's financial information, these institutions could provide a wider range of financial service offerings, leveraging their existing relationship with the customer.

Other attempts to address these problems have included various incantations of a single viable virtual banking model called a value network: a value network is formed when banks and other financial services providers collaborate to offer their customers comprehensive packages of financial services, which are delivered through branch offices and various electronic outlets. For individual financial services customers, value networks represent a much higher level of service because financial services can be highly personalized and delivered with unprecedented speed.

However, no single component of the financial services industry is in a position to create a value network that all may use, thus a single value network system. Each wants to "own" a proprietary network and exclude others in a manner that prevents erosion of their customer base. Further, each component of the financial services industry would simultaneously like to leverage their customer base by offering new services traditionally provided exclusively by the other financial service providers. As a result, the successful construction of a single value network system must come from a third party provider with a comprehensive, customer-focused vision and a commitment to equal access for all the financial services providers within the financial services industry.

It is therefore desirable to provide a modular and open value network system that can communicate with conventional systems to take full advantage of existing technologies. Such a system should be able to run multiple financial applications and process financial transactions and financial information on a single network-oriented system.

It is also desirable for such a system to perform concurrent real-time banking, brokeraging, clearance, analytics, and additional functions. It would be especially advantageous if such a system could use a multiple-tier, open program applications software and hardware architecture that enables access to and delivery of multiple financial services.

II. SUMMARY OF THE INVENTION

The present invention is directed to a method and system that obviates problems due to the limitations and disadvantages of the prior art.

It is an object of the invention to run diverse financial service applications on a value network.

It is another object of the invention to deliver financial services at a reduced cost relative to traditional financial services delivery systems.

It is still another object of the invention to be operational in a baseline configuration sufficient to run applications in time to capitalize on the Y2K problem.

It is a further object of the invention to have extensible interfaces to most current and future financial service hardware and software client types.

It is still a further object of the invention to facilitate the seamless interaction with financial services providers.

It is yet another object of the invention to meet and exceed existing security standards of the financial services industry.

It is also an object of the invention to ensure financial transaction integrity.

It is additionally an object of the invention to be well-documented and well-understood by its designers, builders, operators, managers, and owners.

It is still also an object of the invention to be universally accessible and always available for use.

To achieve these and other objects, and in accordance with the purposes of the invention, as embodied and broadly described, one aspect of the invention includes a method of organizing a value network of a plurality of data centers, comprising the steps of establishing a network center, connecting the network center to the data centers by a high-speed network, redundantly storing data in each data center, and providing communication with the data centers via a wide area network.

In another aspect, the invention includes a method of organizing a presentation layer for a value network executed by a data processor, comprising the steps of communicating with a client

on-line, receiving a batch file from a client, receiving a decision support request from the client, obtaining data from a client, and presenting a response to the client.

In yet another aspect, the invention includes a method a method of organizing an application logic layer for a value network, executed by a data processor, comprising the steps of communicating with a client in the network using middleware, authenticating the client with a security database, processing a request by the client pursuant to an application process, and replicating the request over multiple communication channels.

In still another aspect, the invention includes a method of organizing a data access layer for a value network, executed by a data processor, comprising the steps of storing data for utilization by the network, interfacing with the data access layer to retrieve stored data, queuing a process for implementation by the network, and maintaining a historical archive of images for processing by the network.

In a further aspect, the invention includes a method of organizing a data center on a value network, comprising the steps of providing an application server for applications used by the value network, providing a replica application server for applications used by the value network, providing a database server for data used in the value network, providing a replica database server for data utilized by the value network, providing a communications server for the interconnecting of servers, and connecting the servers over a network.

In an additional aspect, the invention includes a method of processing a message by a client over a value network, executed by a data processor, comprising the steps of initiating a request by the client to join the network, submitting a message for an application, routing the message to an

application server, invoking the message by the application server, and verifying a status regarding the condition of the message at the application server.

Additional aspects of the invention are disclosed and defined by the appended claims. It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory and are intended to provide further explanation of the invention as claimed.

III. BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate systems and methods consistent with the invention and, together with the description, serve to explain the principles of the invention.

In the drawings,

Fig. 1 is a schematic representation of conventional financial services;

Fig. 2A is a schematic representation of the Flobal Electronic Financial Services (GEFS) concept;

Fig. 2B is a schematic representation of the overall architecture concept of a GEFS system;

Fig. 3 is a block diagram showing the physical architecture of the GEFS system, as shown in Fig. 2B;

Fig. 4 is a block diagram showing the processing components of the physical architecture of the GEFS system, as shown in Fig. 2B;

Fig. 5 is a block diagram showing the logical architecture of the GEFS system, as shown in Fig. 2B;

Fig. 6 is a more detailed block diagram showing the logical architecture of the GEFS system, as shown in Fig. 5;

Fig. 7 is a schematic representation of an enterprise wide area network (WAN) implementing the logical architecture of the GEFS system, as shown in Fig. 6;

Fig. 8 is a schematic representation of data center redundancy for the WAN, as shown in Fig. 7;

Fig. 9 is a schematic representation of frame relay and a virtual private network, as utilized by the WAN in Fig. 7;

Fig. 10 is a schematic representation for the extranet connections, as shown by the WAN from Fig. 7;

Fig. 11 is a schematic representation of the types of clients serviced by the extranet connections, as shown in Fig. 10;

Fig. 12 is a schematic representation of the domain structure for a middleware application, as shown in Fig. 5;

Fig. 13 is a block diagram of an application running on parallel database (PDB) clusters;

Fig. 14 is a block diagram showing how an application running on PDB clusters recovers from a RDBMS failure;

Fig. 15 is a block diagram showing how an application running on PDB clusters recovers when, in addition to an RDBMS failure, the node running both administrative and application server processes experiences an outage;

Fig. 16A is a schematic represenation of the interaction among replication services;

Fig. 16B is a schematic representation of the interaction among replication servers, upor failure of a data center;

Fig. 17 is a schematic representation of the interaction among replication servers, upon recovery of a failed data center;

Fig. 18 is a schematic representation of the three major areas of operations and administration within the GEFS environment;

Fig. 19 is a schematic diagram showing the GEFS monitoring architecture of the GEFS system;

Fig. 20 is a block diagram showing the data replication system utilized by the GEFS system;

Fig. 21 is a block diagram showing a detailed view of a data center for a network database;

Fig. 22 is a block diagram showing a preferred implementation of a database with two access machines;

Fig. 23 is a schematic representation of conflicting application requests for access to data from the same block, and how such requests are intercepted by the distributed lock manager (DLM);

Fig. 24 is a schematic representation of the recovery steps taken by a database network upon an instance failure;

Fig. 25 is a schematic representation showing a database network connection to the database through middleware;

Fig. 26 is a schematic representation of data center redundancy for a WAN;

Fig. 27 is a schematic representation of the redundancy of communication links among multiple data centers;

Fig. 28 is a schematic representation of two data centers with the normal operational flow of transactions through frame relay clouds;

Fig. 29 is a schematic representation of two data centers, where the flow of transactions through frame relay clouds has been interrupted by a data center failure;

Fig. 30 is a flow diagram showing the interaction between a client and the GEFS system;

Fig. 31 is a schematic representation showing the hierarchy for the security architecture of the GEFS system;

Fig. 32 is a schematic representation of the Jailbreak application module showing the architecture of the server infrastructure;

Fig. 33 is a schematic representation of a customer-centric data model, as used by the Jailbreak application module;

Fig. 34 is a schematic representation of the clients available under the Jailbreak client computing architecture;

Fig. 35 is a schematic representation of the software stack for a JavaStation;

Fig. 36 is a schematic representation of the software stack for a PC; and

Fig. 37 is a schematic representation of a smart card application module integrating with the architecture of the GEFS system.

IV. DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

A. Overview

B. Network Architecture

1. Concept Architecture

a. Layers of Network

i. Presentation Layer

ii. Application Logic Layer

iii. Data Access Layer

b. Preferred Features of Concept Architecture

2. Physical Architecture

a. General Technical Criteria

b. Specific Technical Criteria

i. Structure of Physical Architecture

ii. Processing by Physical Architecture

c. Preferred Features of Physical Architecture

3. Logical Architecture

a. Structure of Logical Architecture

b. Processing by Logical Architecture

c. Preferred Features of Logical Architecture

C. Network Middleware

1. Technical Criteria

2. Operation

3. Operation Scenario

D. Network Software

1. GEFS Areas of Operation

a. System and Network Monitoring

b. System and Network Administration

c. Change Control

2. Preferred Features

E. Network Database

1. Overview

2. Preferred Architecture

3. Preferred Features

4. Redundancy

F. Clients and Outside Systems

1. Client Computing

a. Overview

b. Interfaces

i. Clients

ii. Outside systems

iii. Other Interfacing

iv. Technical Criteria

v. Categories of Client Computing

c. Preferred Interfacing

2. Java Computing

a. Overview

i. Technical Criteria

ii. Architectural Principles

G. Security

1. Requirements

2. Security Architecture Components

a. Machine Security

b. Network Security

c. Application Security

d. Authentication Security

e. Physical Security

f. Enterprise Security

H. Application Module: Jailbreak

1. Technical Architecture

2. Logical and Physical Overview

3. Hardware

4. Database

5. Network

   a. Mini-Branch Considerations

   b. Standard-Branch Considerations

   c. Premium-Branch Considerations

6. Client

   a. Technical Criteria

   b. Overview

   c. Java Considerations

I. Application Module: Smart Card

   1. Feasibility Analysis

   2. Overview

A. Overview

Reference will now be made to preferred implementations of the systems and methods consistent with this invention, examples of which are shown in the drawings and described below. Where appropriate, and as recited herein, the reference numerals refer to the same or similar elements. The claims define the scope of the invention, however, not the following description.

A value network consistent with a preferred embodiment of this invention runs financial applications and processes financial transactions and financial information. The network provides for the exchange of financial transactions and financial information between different financial applications and financial service providers accessing the network. Because the network efficiently runs financial applications, processes financial transactions, and transmits financial information by

means of accessibility, salability, and high security, the network makes an unlimited range of financial services available.

A preferred embodiment of this invention includes the Global Electronic Financial Services (GEFS) system, a state-of-the-art information technology framework upon which financial services application modules will be deployed on a three-tier value network. The core of the GEFS system is an information technology framework that is infinitely scalable, always available to an unlimited client pool, and comprised of open, best-of-breed technologies. To implement this, the GEFS system encompasses certain operating design goals, including cost-effectiveness, rapid deployment, extensibility, flexibility, reliability, risk control, and treatment of information as a key asset. As a result, the GEFS system offers a host of advantages over the present systems for running financial applications and processing financial transactions and financial information, including modularity, an open architecture, greater efficiency, complete compatibility, open-endedness, ample connectivity, ready receipt of updates, and security.

The GEFS system is modular. In the GEFS system, financial transaction services are available through individual GEFS application modules, which will be sold to those customers who choose a module to "plug into" the GEFS system. The GEFS network also allows financial services providers to plug into this GEFS system, as if the GEFS system were a financial utility. Thus, much as an electric or gas utility makes services available universally and allows users to draw on the utility's resources to the degree they are desired or required, the GEFS system makes a broad range of financial services processing options available to those who choose to plug in, allowing as many services as are desired or required to develop the customer base of each user. The modularity of the GEFS network architecture enables effective, focused development and implementation of new

financial applications. Further, individual business needs can be identified, designed, developed and implemented as functional solutions. Thus, the modularity of the GEFS system leverages its existing workstation, network, and system and application components while adding additional business functions. Also, the GEFS system provides for connecting existing legacy information technology systems with the GEFS network, leveraging the rich functions and previous investment in such systems.

The GEFS system is open-ended. Using both an open architecture and a open programming interface, the GEFS system operates in a global, secure, multi-user, scalable client-server platform environment. The open architecture includes an open application interface to assist the development of new financial applications and to make it easier to port existing financial applications on legacy systems. The GEFS network thus accommodates existing as well as newly developed financial applications, thereby providing consolidation of data and efficient management of network resources. The GEFS network also provides access to financial services and applications and allows processing of large volumes of financial transactions among a large number of users.

The GEFS system is efficient. Because of the distributed nature of the overall GEFS architecture, using a high-availability and centralized management as well as an optimal performance network, the GEFS network provides reliable, fast, and extensible services at all points of use. Given the rapid growth planned for the network infrastructure of most users, all components of the GEFS system are designed to be highly extensible for purposes of network breadth and bandwidth.

The GEFS system is compatible. The GEFS system consolidates business and other financial transaction information from a wide array of financial applications. Due to this compatibility, the

GEFS system allows review and analysis of the information and data stored in the network to provide comprehensive results that include information and data collected from each data source. Thus, the GEFS system enables financial service providers to interact over a computerized network and extend an immediate presence into electronic commerce, including support for ATMs, telephone, Internet, and other PC-based access capabilities. Due to this compatibility, the GEFS system enables rapid development and deployment of additional, integrated financial applications.

The GEFS system is cost effective. The architecture and methods of the GEFS system engine provide cost-effective delivery of financial applications and services. In particular, the GEFS system breaks from existing pricing practices by offering more financial services for less cost to the user by allowing multiple financial applications to run on the GEFS system. This greater functionality creates efficiencies in managing only a single set of network resources. Furthermore, the use of new technologies by the GEFS system promotes quick and efficient development of business solutions based on new networks and object/component implementations, also resulting in a cost advantage because the GEFS network inexpensively takes advantage of the network engine's features and functions.

The GEFS system is open-ended. All of the GEFS operating systems, applications, database components, and client service components enable the integration of a large array of inexpensive hardware configurations, such as less expensive central servers, database components, network components, and client configurations. Due to this flexibility, the GEFS architecture is entirely open-ended, which enables the easily implementation of new technology as it becomes available.

The GEFS system provides for greater connectivity. One of the largest costs to financial service providers and retail banks is the use of restrictive point-to-point WANs. Generally, WANs

enable bank branch locations to interconnect their workstations directly to service providers, but point-to-point WANs are restrictive and inflexible. The GEFS network architecture uses new frame relay dedicated circuit connectivity, or virtual private network (VPN) capabilities, to interconnect GEFS network users, including bank branches, more robustly and much less expensively.

The GEFS system enables timely and efficient updates. The GEFS network engine is a powerful, yet stable, business platform. This stability extends to security, redundancy and availability. Regardless of the client or access network, the GEFS network can ensure that each transaction is completed successfully. Components, such as, the BEA TUXEDO® by BEA Systems, Java's Jolt, and Connectivity products, provide many of the GEFS features. The Oracle® relational database and the Sun® system components are designed and configured to work in conjunction with. the other GEFS network components to allow the GEFS network to manage information securely while providing high performance and availability. Nonetheless, the modular software and hardware architecture of the GEFS system has the flexibility to enable users and other financial service providers to improve the services they provide, by updating these component features easily.

The GEFS system ensures security. The GEFS system contains controls for secure access. Nonetheless, the GEFS system architecture is designed and maintained to ensure that information delivery remains effective for a wide range of delivery channels beyond traditional bank branch facilities. The GEFS system facilitates these new delivery channels, including PC-based workstations, Java-enabled Internet or internal network access, call centers, smart cards, and ATMs.

B. Network Architecture

Figure 2A is a schematic representation of the GEFS concept. GEFS concept 100 includes infrastructure 110, which contains the GEFS architecture. GEFS infrastructure 110 will

accommodate GEFS applications 115. GEFS applications 115 include an unlimited number of application modules. GEFS applications 115 include module 140. This module entails banking functions that include services primarily for tier 2 and tier 3 banks, for accomplishing core retail banking functions, commercial banking functions, and wholesale banking functions now being provided by service providers. GEFS applications 115 also include module 120. This module entails direct banking functions, which are to be provided primarily to consumers of the tier 2 and tier 3 banks and other financial services providers. GEFS applications 115 further include module 130, or the electronic compliance reporting module. As shown in Fig. 2A, other financial applications, or modules, may also be plugged into global financial utility 150.

1. Concept Architecture

Figure 2B is a schematic representation of the architecture concept of GEFS system 200. GEFS system 200 preferably operates as a three-tier network. Key components of the GEFS architecture include: a small footprint; non-proprietary client language to facilitate distribution of presentation layer software components to a wide range of client types; a robust transaction processing monitor middleware platform to encapsulate application logic as services, manage access to system resources, and guarantee the ACID (atomicity, consistency, isolation, and durability) properties of transactions; a proven, relational database management system to manage data access efficiently in a highly standardized manner; a powerful, multi-threaded operating system to run servers and realize maximum use of hardware resources; symmetric multi-processing machines for servers to ensure process-level scaling; massively parallel or clustering technology to ensure redundancy and peak transaction load performance; high bandwidth; and virtual and dedicated redundant networks running standardized communication protocols to ensure communications

connectivity and performance between tiers. As described below, the three tiers of the three-tier network relate both to the hardware architecture and the software programs.

As shown in Fig. 2B, GEFS system 200 includes GEFS network 205, which represents a fully scalable, flexible, accessible, and secure financial transaction engine. GEFS network 205 routes financial transactions or financial information over a computerized network system. The messages originate from clients at presentation layer 220 and are transmitted to application logic layer 240, which contains servers containing logic distributed over the network, and processes and records the messages at data access layer 280. Servers at application logic layer 240 are linked to databases at data access layer 280 and provide data to middletier application servers at application logic layer 240.

a. Layers of Network

As recited above, and as shown in Fig. 2B, GEFS system 200 includes three distinct layers: a top tier, called presentation layer 220, having front-end, thin clients; a middle tier, called application logic layer 240, having application servers; and a bottom tier, called data access layer 280, having network data centers and decision support modules. GEFS system 200 may also interface with additional information services providers that can port information into the network and link the network to technology partners, such as additional financial services providers. Data centers connect with the middle tier application servers, which typically include BEA tuxedo servers. Data centers and application servers may also maintain links with outside information services, such as Fedwire, ACH, Chips, and Swift. GEFS system 200 can also link directly with existing outside service providers or provide an interface to any other existing legacy system. The data centers contain the financial information, typically stored in Oracle® relational databases.

i. Presentation Layer

Presentation layer 220 is a thin layer that provides the functions of data collection, request initiation, and response presentation. Presentation layer 220 contains neither business logic nor a data access instruction set. As shown in Fig. 2B, presentation layer 220 supports three types of clients: on-line clients 225, bath clients 227, and decision support system (DSS) clients 229.

On-line clients 225 support transactions executed in real-time. These transactions have the most stringent response time requirements. Examples of on-line clients are bank teller workstations, voice response units, smart cards, and Internet clients.

Batch clients 227 include clients who process files containing transactions. These clients are less concerned with individual transaction response time and more concerned with system. throughput and recoverability from partial failures. ATM file processing is one example of a batch client.

DSS clients 229 include clients who submit requests that have lengthy execution times. The clients often have large and unpredictable amounts of data.

ii. Application Logic Layer

Application logic layer 240 contains the business rules for transaction processing but does not include details of data schema organization and data access details. Application logic layer 240 contains seven types of application processes: web processes 242, security processes 245, communication processes 248, transaction processes 251, DSS processes 257, replication application processes 254, and external gateways 260.

Web processes 242 act as a repository for client Java applets, which will be downloaded to client browsers as per user requests.

Security processes 245 contain and apply logic for authentication of client sign-in requests to GEFS system 200 and also manage the security database.

Communication processes 248 act as connection points from clients and route connection requests to the appropriate application server.

Transaction processes 251 contain the application logic that represents the systemic implementation of the business processes the application is designed to support.

DSS processes 257 provide DSS services for ad-hoc and long running queries.

Replication application processes 254 provide data duplication services across multiple nodes, ensuring durability of data in the event of localized failures or disasters. Replication application services are of two types: on-line and queued. A given on-line service will call the corresponding replication service in its fail-over data center to update the remote operational database within the scope of the same transaction. Queued replication services will queue requests in a TP monitor queue facility, and subsequently, a queue reader will periodically drain this queue and request the corresponding replication service on the fail-over data center.

External gateways 260 provide connectivity to external systems. There are two types of external gateways: incoming and outgoing. Incoming gateways receive a transaction request from an external system, such as, for example, ATM 261 and call center 262, and submit the transaction request to GEFS system 200. Outgoing gateways, such as, for example, Fedwire 267, submit requests to external systems. Outgoing gateways are of two types: online and batch. On-line outgoing gateways are called during transaction processing. On-line outgoing gateways submit requests to external systems in real time, wait for a response, and pass the response back to an application server. Batch gateways, such as, for example, NUCHA 264, are triggered at pre-defined

times. Batch gateways obtain their input from database tables, submit requests to external systems, collect responses from external systems, and store responses in a database table or flat file.

iii. Data Access Layer

Data access layer 280 facilitates access to data stores and insulates upper layers from data schema details. Data access layer 280 also contains databases 290. The GEFS data access layer contains four types of data access servers which communicate with databases 290: data processes 282, queue processes 284, image processes 286, and historical processes 288.

Data processes 282 contain the access logic and data schema details and provide access to the physical data store.

Queue processes 284 provide an interface to the TP monitor queue facility.

Image processes 286 provide an interface to imaged data stored in either an RDBMS database or an optical device.

Historical processes 288 manage a store of aged data before the data is archived.

b. Preferred Features of Concept Architecture

GEFS system 200 provides for many particular features, including scalability, high capacity, high throughput, low response time, manageability, accessibility, availability, interoperability, usability, ease of development, robustness, data integrity, security, and internationalization. Another element of GEFS system 200 and GEFS network 205 is cost effectiveness. Providers of financial services have traditionally charged exorbitant service fees. Systems and methods consistent with this invention can provide a lower cost solution to its customers. Accordingly, using GEFS system 200, a well-conceived and well-implemented information technology framework is provided for a competitive cost advantage in an increasingly competitive marketplace.

Scalability describes the capability of a comprehensive system, application, and network environment to sustain growth over time without affecting critical processes. This feature of scalability, for example, provides the ability to add or change processing power, or to configure software applications seamlessly to handle load changes. There are four major types of scaling in a client server architecture: vertical scaling, or scaling by adding systems, such as adding machines to a cluster; horizontal scaling, or scaling by adding resources, such as adding CPUs or memory to a system; replication scaling, or creation of another entire environment, such as adding another data center; and upgrade scaling, or change of hardware platform, such as exchanging one server for a much more powerful server. GEFS system 200 is infinitely scalable because the system will support multiple application modules. The system allows for these modules to be implemented in a staggered sequence with respect to other application modules and incrementally to the target customer base of the user for that module. Therefore, GEFS system 200 is architected to guarantee the scalability necessary to achieve the target growth of the user.

Capacity describes the ability to support large volumes of data, multiple users, and high volumes of batch or on-line transactions. Adequate capacity enables the sharing of information resources among many requesters while ensuring that the performance and availability demands of peak workloads are supported. GEFS system 200 has high capacity because the system supports large volumes of data, transactions, numbers of clients, and geographic distribution. Indeed, GEFS infrastructure 110 also has the capacity to handle large hardware components and other processing equipment.

Throughput describes the sum of all transactions across all application modules and machines that a system is capable of handling in a given time period. In a preferred implementation of the

application module for the core retail banking functions (i.e., the Jailbreak module), GEFS system 200 handles over 2 billion transactions per year, which is only a subset of the total number of transactions that will ultimately flow through the system. After all, the minimum throughput of GEFS system 200 running a complete suite of financial application modules is expected to be at least 30 billion transactions per year. GEFS system 200 has the throughput to handle such numbers of transactions.

Response time describes the perceived time it takes for a system to respond to a request. A well-designed online system can be expected to provide a response time that will serve the business needs of the business for which it is designed. GEFS system 200 ensures proper response time by specifying the performance requirement for all critical transactions and conducting performance tests for every application module.

Manageability describes the ability to effectively monitor, manage, and reconfigure any component of a system. In the context of a network, manageability is the ability to effectively manage the following: system software, network hardware, each application and its performance, components of the system, and changes and enhancements to these components. In a preferred implementation, GEFS system 200 comprises a large distributed enterprise with over 60,000 nodes. Smooth, efficient operation of this system includes a high degree of manageability for every component device.

Accessibility describes the aspect of a system that makes it possible for people and other systems to use it without respect to geography or other logistical constraints. GEFS system 200 supports information access for numerous external systems including existing legacy information

investments, ATM networks, and trading systems, many of which are not confined to a specific geography or single facility of operation.

Availability describes the capability of a system to continue to function without constraining its users, even for component failure. Because consumers expect to be able to conduct their business whenever they choose, and without restriction, GEFS system 200 must be highly stable, reliable, and therefore, available. This availability is accomplished through combinations of component redundancy, software-controlled fail-over, or system redundancy. Thereby, as recognized in GEFS system 200, the end result of a well-planned high availability solution is low downtime.

Interoperability describes the ability to exchange data and interact with a diverse set of client, server, and database systems in a transparent manner. GEFS system 200 is interoperable because the system integrates with a wide range of legacy information sources, such as the National Automated Clearinghouse Association (NACHA) 264, Society for Worldwide Interbank Financial Telecommunication (SWIFT) 265, ATM 261, and Electronic Data Interchange/Electronic Funds Transfer (EDI/EFT) 266, to provide a true value network. GEFS system 200 is also interoperable because the system makes it possible both to add to or modify the interfaces to an external system with minimum changes, and to integrate new technologies seamlessly.

Usability describes the degree to which a system facilitates its users' efforts to learn how a system operates and how to use the system as intended, quickly and with minimal effort. Usability includes attributes such as human engineering, aesthetics, user friendliness, consistency of user interface, intuitiveness, and standards compliance. GEFS system 200 is useable in this regard. This usability extends to clients, operations, and management capabilities. Due to this usability, as new

capabilities are added to GEFS system 200, customers, consumers, and operators will be able to easily learn to use these new functions.

Ease of development describes the ability to reduce time to market for new products and to produce an easily maintainable product. Ease of development requires that enhanced functionality can be added in the shortest possible time frame with the highest possible quality following industry standard guidelines for software development. GEFS system 200 provides for ease of development.

Robustness means that the components of a system are redundant and flexible. GEFS system 200 processes and maintains information critical to the stability of financial transactions. For this reason, GEFS system 200 includes redundancies and other flexible features so that the failure or destruction of any component of the system will not expose information to loss or preclude continued processing.

Data integrity encompasses atomicity, consistency, isolation, and durability. Atomicity means that all data updates that constitute a transaction are treated as a single unit of work so either all updates occur or none occur. Consistency means that the system preserves internal consistency to satisfy all the referential constraints of the database. Isolation means that any set of transactions that run concurrently will produce the same effect if they run serially. Durability means that once a transaction is committed, updates will not be affected by system failures. GEFS system 200 maintains data integrity.

Security describes the ability of a system to prevent theft, modification, or damage to itself and the information it manages either intentionally or unintentionally, or internally or externally. GEFS system 200 includes and maintains security.

Internationalization describes the ability of a system to function in an international environment. GEFS system 200 incorporates internationalization by making architectural choices that support international considerations, such as language and culture.

2. Physical Architecture

The physical architecture of GEFS system 200 provides reliable, fast, and extensible services at all points of use. GEFS network 205 acts as the primary enabling technology for this architecture.

a. General Technical Criteria

With regard to the physical architecture of GEFS network 205, the distributed nature of the network components promotes scalability, capacity, throughput, response time, manageability, accessibility, availability and robustness, connectivity, interoperability, useability, ease of development, data integrity, security, and internationalization. In addition, for GEFS network 205, in a preferred implementation, all components are highly extensible in network breadth as well as bandwidth capacity.

For scalability, GEFS network 205 uses modular components (e.g., routers, WAN circuits) that can be grown in place to full capacity, and then be augmented by another circuit or piece of hardware. This architecture supports unlimited expansion, because GEFS network 205 must be able to grow rapidly. Accordingly, the network has been designed with large building blocks that can be expanded by configuration, not physical infrastructure. This approach also allows dynamic addition of bandwidth on the WAN by reconfiguring existing lines, typically initiated by a phone call, thus allowing for on-line reconfiguration of the communications equipment within hours, not days. In addition, GEFS network 205 uses the most advanced and proven network technologies. As emerging

technologies mature and prove themselves, the architecture allows both for addition or replacement of LAN and WAN transports without major disruptive changes to the network.

For capacity, GEFS network 205 supports an architecture that allows for the unlimited addition of capacity and circuits, as needed.

For throughput, GEFS network 205 has network provider contracts that include provisions for bandwidth on demand, such as during peak load.

For response time, GEFS network 205 has virtual circuits between data centers and remote sites that ensure that responses are made within acceptable time windows by facilitating dynamic routing.

For manageability, the network building blocks in GEFS network 205 are deliberately consistent and manageable by industry standard interfaces, such as Simple Network Management Protocol (SNMP) and Java Management Application Programming Interface (JMAPI). This enables GEFS network 205 to completely exploit state of the art network management tools to monitor and control the entire network enterprise from central points of control. Using standard interfaces also simplifies deployment and growth of remote offices. These centralized management features and technology give the local clients freedom from local MIS tasks with complete management and monitoring from the data centers.

For accessibility, use of transmission control protocol/Internet protocol (TCP/IP) and frame relay in GEFS network 205 ensure remote sites can be dropped in anywhere.

For availability and robustness, GEFS network 205 has no single point of failure. Instead, there are multiple tiers of redundancy at all circuit and hardware levels. GEFS network 205 is designed with two or three tiers of redundancy at all circuit and hardware layers to insure the

availability of completely autonomous paths between every point. For example, the tiered architecture deliberately includes multiple network providers for each component to avoid single points of failure.

For connectivity, GEFS network 205 also provides end-to-end connectivity between the end user client, such as a banking teller, remote Internet client, or call center attendant, and the data accessed at the data center. This connectivity is reliable and available around the clock. All communications occur intact and securely, and any failures are reflected to the end-user and GEFS operations center immediately.

For interoperability, GEFS network 205 is compatible with a TCP/IP network.

For usability, standard protocols and devices in GEFS network 205 facilitate operator control of the network.

For ease of development, all enterprise tools for development in GEFS network 205 integrate seamlessly with TCP/IP.

For data integrity, all networks are private and isolated to the GEFS network 205. The TCP/IP protocol along with the application layers insure data integrity.

For security, Network Information System Plus (NIS+) will be employed by GEFS network 205 to provide centralized administration and control of wide user access.

For internationalization, services through international providers ensures compatibility by GEFS network 205 with local languages and cultures.

Furthermore, all components of GEFS network 205 are also fully extensible to accommodate evolution of the network. A balance of state-of-the-art and widely-used transports and technologies ensure extensibility without taking unwarranted risks with commercially unproven technology.

Thus, GEFS network 205 accommodates integration with existing legacy components as well as extension to emerging technologies as they prove viable.

b. Specific Technical Criteria

i. Structures of Physical Architecture

Figure 3 is a block diagram showing the physical architecture of GEFS system 200. GEFS physical architecture 300 includes data centers 310 connected through link 315 such as high-speed redundant virtual private links. Each of data centers 310 is configured (in items of machine size, number of machines, disk space, network bandwidth, etc.) to handle its own load and the load of a partner data center. For example, the Jailbreak application module, as described below, has two data centers, each containing the necessary capacity to handle the complete system load individually. The other components of GEFS physical architecture 300 include: GEFS monitoring operation center 320, a bus 345 connected to data center-to-data center Wide Area Network (WAN) connectivity, such as connection 317 via routers 346; data center-to-remote site WAN connectivity 325 via routers 346 and frame relay switches 347; remote site Local Area Network (LAN) connectivity 326 via gateway router 348; and Internet, extranet connections 365 via firewall 349.

GEFS data centers 310 are connected by redundant high speed private networks, such as link 315, which may be T3 or ATM lines. Data centers 310 include a bus 345, generally an Ethernet bus, connected to enterprise database clusters 340, which are connected to enterprise database 330, and further include enterprise application servers 350, monitoring management station 351, router pair 346, and gateway router 348. Data centers 310 may also include application gateway services 355, legacy application gateway services 360, and enterprise application services 365. Application gateway services 355 includes access to such services as ACH, Swiftwire, and ATM networks.

Legacy application gateway services 360 include access to legacy systems at data centers 310. Application gateway services 355 connect to gateway router 348, and thereby access outside legacy networks 327 via extranet connection 326. Enterprise application services 365 includes such services as on-line analytical processing (OLAP) reporting, enterprise compliance, and marketing data marts.

Data centers 310 may interact with Internet and other extranet services by Internet, extranet connections 365. As shown in Fig. 3A, Internet, extranet connection 365 accesses service center 380. Service center 380 contains router 381, Web gateway 382, and application services 383. Service center 380 also allows access to Internet 390 by connection 385. From Internet 390, access is further provided to virtual private access 391, banking clients 392, and clearing clients 393.

GEFS physical architecture 300 also supports isolated topologies for security. A high speed interconnect is constructed between the data centers, such as link 315. Other components of GEFS physical architecture 300 are extensibility and availability for the evolutionary growth of any GEFS system 200. Therefore, using this architecture, state-of-the-art and widely-used transports and technologies insure extensibility without taking unwarranted risks with commercially unproved technology. Further, state-of-the-art in network management tools manage this distributed network as it grows more complex, and its scalability and redundancy leverage directly from its several distinct layered pieces. As the technology matures in any GEFS system 200, for example, other components can be employed in a controlled fashion to supplement the mesh. As an illustration, private virtual connections over public networks can be used as a layer, initially for non-critical traffic, and given more responsibility as the reliability and performance are proven out. Outside

network providers demonstrate such expandability over outside network providers 325, which may connect with other systems 370, call centers 372, or other input devices 374 (e.g., kiosks).

ii. Processing by Physical Architecture

Figure 4 is a block diagram showing the basic processing components of GEFS network 205. As shown in Fig. 4, GEFS data centers 405 include multiple data servers 410 linked to GEFS application server clusters 420 and other information services providers 425. In a preferred implementation, data servers, enterprise clusters, and other components of GEFS data centers are linked by a fast Ethernet bus, such as one on the order of a gigabit, such as bus 430. Bus 430 connects data centers 405 with other application gateways 450 through middleware 440. Application gateways 450 allow application server clusters 420 and other information services providers 425 to share data and the network infrastructure with outside systems 460. In addition, other connections 470 may provide more direct access to inside systems 480 or information links 490. Inside systems 480 may include service providers 484 and existing legacy systems 486.

Bus 430 connects to routers (not shown) that transfer incoming and outgoing messages to data centers 405. As shown in Fig. 3, data centers 310 connect with the Internet, extranet service centers 380 through firewalls 349 via routers 346. The service centers typically include their own routers and busses for connecting to on-line banking web services, virtual private banking, and elite private banking, such as via routers 381. The Internet also allows access to browser-based clearing web services. The routers also direct messages from public Internet providers for connecting with other remote banking front-end clients. Thus, the firewall provides a layer of security to control access to GEFS network 205 from public sources.

Bus 430 also connects data centers 405 to outside systems 460. As shown in Fig. 3, outside network providers provide redundant frame relay clouds 328 to insure continuous operation even if one network provider is down. Redundancy is discussed in more detail below. In sum, however, multiple call centers 372 may connect to multiple frame relay clouds 328 and multiple input devices 374, and multiple GEFS data centers 310 connect to each other over links 315.

Normally, again as shown in Fig. 3, each pair of data centers 310 services clients in its geographical area, but when a remote data center fails, the stable data center will service the clients of its failed partner data center as well. In a preferred implementation, as described in more detail below, pairs of data centers 310 (or data centers 405 as shown in Fig. 4) contain clusters of Enterprise class Sun machines running under Sun's High Availability framework for the applications and BEA's HA TUXEDO framework for the middleware. Therefore, each machine and each functionality will have a backup machine and a backup functionality with a shared/mirrored disk.

iii. Preferred Features of Physical Architecture

In a preferred implementation, GEFS system 200 utilizes the Sun Microsystems UltraSparc architecture. This 64-bit architecture is widely supported as the primary porting platform for most enterprise software vendors. The 64-bit architecture allows for very large memory addressing, which directly leverages outstanding database and computing performance. The Enterprise class of servers are commercially proven and ongoing performance upgrades are provided regularly. The Enterprise class of servers thus are highly extensible, and can easily be integrated into a multi-server distributed environment.

In general, the Enterprise class servers are extensible at several levels. The number of processors can be upgraded and downgraded as a quick maintenance task, and can range between 1 and 64. The processor boards are available in various performance levels (170/200/250/300Mhz). The various models of Enterprise servers have generally interchangeable parts. Furthermore, the Enterprise Solaris operating system allows transparent migration of software within the entire Ultra hardware line, also demonstrating this extensibility. Software can be developed on Ultra workstations and deployed or served to Enterprise servers without modification. Also, applications can be moved to larger or smaller machines as needed on demand.

In particular, the Enterprise 10000 from Sun Microsystems is the hardware platform of choice for GEFS. These machines can operate at approximately 1348 transactions/second, and four of these machines in a cluster support the load and scalability requirements required for GEFS applications. Sun's Enterprise 10000 hardware, and the gigabit Ethernet networking card, are the server platform and networking interface standards because of their capability to handle these large transaction loads and their capacity to provide the networking support required to handle the user workload and the geographic distribution of resources. The Enterprise 10000 is currently Sun's top-of-the-line machine, provides an overall system bandwidth of 12.8 gig/sec, and provides the fastest interconnect in its class for clustering. For GEFS applications, these machines are deployed in clustered configurations to provide the necessary capacity.

As shown in Fig. 3, each pair of data centers 310 contains one or more of the following items: enterprise database clusters 340, enterprise application servers 350, and Internet/extranet connection 365. In a preferred implementation, these include a database BEA TUXEDO server pair; a BEA TUXEDO server pair; and a Web and interface server, respectively.

As for the BEA TUXEDO database server pair, the pair of machines in the Database/BEA TUXEDO server pair primarily act as database servers running under the PDB environment. These machines also run BEA TUXEDO servers. TUXEDO services running on these servers require frequent and relatively large interaction with the database.

As for the BEA TUXEDO server pair, the pair of machines in the BEA TUXEDO server pair run BEA TUXEDO application servers and communication servers that handle connections from clients.

As for the Web and interface server, this server runs a Web server and serve up all applets requested by clients. The server may also run gateways for external interfaces, such as interfaces to ATM, NACHA, SWIFT, etc.

Notably, both the database clusters and the application servers use the BEA TUXEDO system, which offers several features that facilitate the building of very large applications. The domains feature of BEA TUXEDO allows one BEA TUXEDO application environment to interact seamlessly with another while maintaining separate administration domains. Using BEA TUXEDO's capability to distribute applications and data, make use of database and system resources efficiently, and provide load balancing capabilities the GEFS Infrastructure provides required capacity.

To coordinate data centers 310, there is GEFS network 205. GEFS network 205 uses an Oracle parallel server (OPS) for high availability, to allow the same database to be accessed from multiple network nodes. The databases are designed so that database servers from different nodes do not use too much shared data in normal operation mode to avoid distributed lock contention. In addition, middle tier application servers can connect over any existing network to front-end bottom

tier devices, such as ATMs, PCs, Java workstations and network computers. Additionally, network computers may connect to GEFS network 205 via a LAN network connection.

Based on the descriptions provided above, the architecture of GEFS system 200 is recited pursuant to physical architecture 300. However, the architecture recited in physical architecture 300 contains other significant features, including scalability, capacity, throughput, response time, manageability and accessibility, availability, interoperability, useability, ease of development, robustness, data integrity, and security.

For scalability, physical architecture 300 uses Sun Microsystems Enterprise Class servers, which can be scaled both horizontally and vertically, by adding systems to a cluster or adding system resources.

For capacity, physical architecture 300 also uses Sun Microsystems Enterprise Class servers that use Symmetric Multiprocessing (SMP) architectures and clustering technologies to handle a large volume of transactions.

For throughput, physical architecture 300 uses fully configured Enterprise servers that can execute more than 1300 TPC-C transactions/sec each.

For response time, physical architecture 300 uses Gigabit speed backplanes, with SMP architecture.

For manageability and accessibility, physical architecture 300 uses Enterprise servers, which are fully Simple Network Management Protocol (SNMP) manageable.

For availability, physical architecture 300 uses clustering/HA/PDB technologies.

For interoperability, physical architecture 300 uses Sun Microsystems Servers which are fully comparable with TCP/IP networking, Oracle DBMS software, and BEA TUXEDO software.

For usability, physical architecture 300 uses a Solaris/Enterprise server, an open system providing system management, and monitoring support.

For ease of development, physical architecture 300 uses Sun servers, the most widely used UNIX based servers on the market providing a comprehensive selection of tools, compilers, and debuggers are supported and available.

For robustness, physical architecture 300 uses redundancy within the box.

For data integrity, physical architecture 300 uses mirrored disks, supported by parallel server software and high availability software.

For security, physical architecture 300 uses a Solaris/Enterprise server supporting numerous security software implementations.

3. Logical Architecture

Figure 5 is a block diagram showing the logical architecture of GEFS system 200, corresponding to physical architecture 300. In particular, GEFS logical architecture 500 includes the logical architecture for the components of a single data center 510, one of the two data centers contained within a pair of data centers 310. For perspective, Fig. 5 also depicts other data centers 515 and network management center 517. Also, Fig. 5 depicts a connection between data center 510 and other data centers 515 by communications link 520. Furthermore, also for perspective, Fig. 5 depicts frame relay network 530, which connects with web server/firewall 32, client networks 534, call centers 536, and other GEFS customers 538. Web server/fireball 532 is further connected to Internet clients 539.

Data center 510 presents a detailed depiction of the logical architecture for a data center. As shown in Fig. 5, data center 510 is connected to the GEFS system 200 via communication equipment

542. The components of data center 510 are then connected by bus 540. The components connected by bus 540 include dual local applications server 545, dual local data server 550 that services database 552, dual replication application server 555, dual replication data server 560 that services replication database 565, local image storage 570 that services image database 572, local historical data server 575 that services long-term database 577, network management station 580, gateway hardware 585, and printer services 598. In addition, two other possible components connected by bus 50 include replication image storage 595 and replication historical data server 596.

a. Structure of Logical Architecture

Figure 6 is another block diagram showing data center 510 in still more detail. Data center 600 includes application server pair 610, database server pair 620, image server pair 630, and historical server pair 640, all of which are connected by bus 650. Application server pair 610 includes local application server pair 613 and replica application server pair 616. Database server pair 620 includes local data server pair 623 and replica data server pair 626. Image server pair 630 includes local image storage servers 633 and replica image storage servers 636. Historical server pair 640 includes local historical data servers 643 and replica historical data servers 646. The preferred implementations are described below.

Application servers pair 610 run the GEFS data applications, using BEA TUXEDO application programming interfaces. These servers are configured as a high availability pair using BEA TUXEDO HA software. Two types of application servers can be found in each data center: local application server pair 613 and replica application server 616. Replica server 616 provides a replica for the data of another data center's local server pair. The characteristics and installed

software on these servers include BEA TUXEDO transaction processing, BEA TUXEDO HA high availability software, transaction routing and control for data servers, and high transactions rates.

Database server pair 620 provides on-line data services within the GEFS applications. Two types of data servers can be found in each data center: local data server pair 623 and replica data server pair 626. Local data server pair 623 contains the local data. Replica data server pair 626 provides a replica for another data center's local server pair 623. These servers are configured as an Oracle Parallel Server pair and include a Solaris 2.5.1 Operating System, Oracle DBMS/Oracle Parallel Server software, BEA TUXEDO software, large databases, and high transaction rates.

Image server pair 630 contains multimedia images of any form of financial records (.e.g, signed contracts, canceled checks, and account statements). This type of information is stored for a medium term. Two types of image storage servers can be found in each data center: local image storage servers 633 and replica image storage servers 636. Local image storage servers 633 contain the local images. Image storage servers 636 provide a back-up for another data center's local image storage server 633. These servers include multimedia storage, Oracle DBMS/Oracle parallel server, BEA TUXEDO software, large size data elements, large quantities of data and low transaction rates.

Historical server pair 640 contains long term records of financial transactions for later retrieval. Two types of historical data servers can be found in each data center: local historical data servers 643 and replica historical data servers 646. Local historical data servers 643 contain the local historical data. Replica historical data servers 646 provide a back-up for another data center's local historical data server 643. These servers include Oracle DBMS/Oracle parallel server, BEA TUXEDO software, large quantities of data, and low transaction rates.

In addition to these servers, data center 600 may also include one or more remote site backup servers 660 (not depicted) for the remote site bank branch servers. There may be many of these systems in order to support the large number of remote sites required by GEFS system 200.

Figure 7 is a schematic representation of an enterprise WAN implementing the logical architecture of GEFS system 200. As shown in Fig. 7, GEFS data centers will house application servers, the overall virtual GEFS database, connections to external data sources (e.g., Swiftwire, ACH, etc.), and centralized management (i.e., the GEFS network operation center). The WAN will need to interconnect several data centers and service traffic between individual remote site facilities and any one of the data centers. The actual bandwidth required between the data centers will be driven by the application requirements. The current architecture will provide for a potential range of bandwidth from T1 (1.44Mb/set) to T3 (144Mb/set) and upwards.

Pursuant to GEFS logical architecture 500, connection of data centers will use a two or three layer, star-meshed network. The actual network transports depend on the bandwidth requirements and best capabilities of the providers in the data center geography. Possible data center WAN connections include: (1) three redundant Cisco 75xx routers, with CiscoWorks HA software; (2) three physically separate OC3 (Optical Channel type 3) fiber-optic cables into the facility; (3) at least two point-to-point connections to every other data center; (4) at least two point-to-point connections to each remote site; and (5) point-to-point connections distributed to ensure no single point of failure. To illustrate this usage and increment, the following table depicts typical data center T3 usage and increment for the connection of one data center to another data center.

	Initial	Growth Increment [addnl OC3]
Number of OC3 cables/data center	3	+1
Initial Bandwidth per channel for data center-to-data center	5Mb/sec [expandable to 44Mb]	increments of 1Mb up to 44Mb
Number of channels per data center	9	+6
Initial Total Bandwidth	30Mb/sec	1Mb up to 144Mb.
Maximum Total Bandwidth	288Mb/sec	+144Mb

Figure 8 is a schematic representation of data center redundancy for the WAN, as shown in Fig. 7. As shown in Fig. 8, an unlimited number of data centers may be connected by one or more means of communication. The preferred implementations are described below.

The data center WAN consists of at least two layers of meshed, high-capacity network, either 73 or ATM. Each point-to-point connection will be extensible from 1 to 144 mb/sec; every additional mesh provider will add up to 144 Mb of extensibility to each link. Each mesh will be provided by a completely different provider to ensure no single-point of failure. Thus, in the event of network-wide failure (e.g., all MFS), connectivity is insured. As an example, as shown in Fig. 8, three distinct service channels are depicted for the WAN, such as service channel A (e.g., a T3 by AT&T), service channel B (e.g., a T3 by MFS), and service channel C (e.g., a T3 by Sprint). Further, the OC3 fiber used by the service channels will contain the following: three distinct Data Service type 3 (DS3) channels, each initially with 5 Mb service from a separate network provider, allowing delivery of 10Mb per OC3 into the data center; each DS3 channel running at 5Mb can be

reconfigured live to deliver a maximum of 44Mb as needed; each OC3 will terminate at a distinct central office; and each OC3 fiber will enter the facility from separate areas.

The service channels are serviced by network providers. Network providers are classified as Tier I, II, and III. Tier I network providers are the largest most pervasive telecommunications providers in the world. Examples of Tier I providers include AT&T, MCI, Sprint, and Worldcom/MFS. Tier II providers are a small group of the largest network integration providers, which use the infrastructure of the Tier I providers to build hybrid, large, multi-vendor networks. Tier I providers also add value by providing network deployment and integration services, as well as collocation services at their network access points. Tier II providers include UUNET, BBN, Genuity, and ICONnet. Tier III providers consist of the remainder of the smaller network providers.

In a preferred implementation of the logical architecture, as represented by the data center redundancy in Fig. 8, GEFS network 205 utilizes a unique tier I or tier II networking provider to avoid single points/vendors of failure and to leverage multiple vendors' disparate network management and infrastructure. This layering of capability provides excellent redundancy in the event of provider-wide performance problems or network outages. Additional layers/providers can be added directly as needed. Furthermore, additional services such as data center services at the network access points may be used at a tier II network provider.

As shown in Fig. 7, an enterprise WAN implements the logical architecture of the GEFS system 200. In contrast to some WAN connections, remote site WAN connectivity involves a reliable and highly available low bandwidth network connection. In general, a given remote site does most of its network traffic with a specific data center, but the remote site will also need to access other data centers transparently for specific tasks and in the event of a catastrophic data center

failure. For that purpose in a preferred implementation, GEFS system 205 uses frame relay and a virtual private network.

Figure 9 is a schematic representation of frame relay and virtual private network, as utilized by the WAN from Fig. 7. The preferred implementations are described below.

As shown in Fig. 9, frame relay technology is used to link all the remote GEFS clients to the central data center. As shown in Fig. 9, a cloud represents a frame relay circuit, into which a data center or remote site has a data port. All ports represented within the cloud have virtual point-to-point access to all other points within the cloud. Frame relay uses the private switched network of I network providers, which routes every frame relay packet dynamically over the best and most available route. Frame relay is widely available in all parts of the USA and most areas of the world. The two-tier mesh of frame relay providers will provide a redundant, highly available, and extensible WAN for data center-to-remote office connectivity. Additional bandwidth can be added on the individual circuits, and new providers can be configured as additional mesh layers as needed. Furthermore, as emerging technologies mature and prove themselves, the present GEFS architecture also allows for the addition or direct replacement of LAN and WAN transports without major disruptive changes to the network.

The benefits of frame relay include economy, reliability, scalability, and mature technology.

For economy, frame relay is priced on port and speed, not distance. Most leased lines are based on distance and cost twice as much as a comparable frame relay circuit.

For reliability, frame relay directly leverages the private network management of the major telecommunications providers. Frame relay circuits dynamically seek the best route and route around problems on the network, whereas a traditional point-to-point network has several single

points of failure, because it uses dedicated wires and circuits. For scalability, a new site can be connected to all other points on a cloud via a single port installation. Thus, no star network needs to be installed for every additional port. In addition, the port speeds of frame relay can be dynamically configured from 56K/sec to 1.44Mb/sec in place. Plus, emerging technology will increase this ceiling to 10Mb/sec. Typically, bandwidth requirements will range from 56K - 128K/sec.

For mature technology, frame relay has been in common use for over ten years and competition has increased the overall quality of private-network based frame relay. Furthermore, provided that any frame network adheres to a quality of service metric known as IETF-RFC 1490, the frame network should guarantee transmission of packets within the committed information rate.

To implement frame relay, Fig. 9 presents a sample implementation of a data center to remote site WAN configuration. Fig. 9 shows two frame relay network clouds interconnecting all data centers and remote sites. The data centers include data center A, data center B, and data center (n). The remote sites include a virtual private network, depicted as a virtual private network cloud, and a call center. The preferred implementations are described below.

The data centers use redundant T3 feeds from each of the two frame relay clouds into each respective data center. These T3 feeds can be placed on unallocated DS3 channels and router connections, as described in the previous section. Notably, the T3 feeds must be distributed as described in the previous section to avoid single points of failure.

The remote sites have a standard Cisco 4xxx series router with two high speed serial interface (HSSI) interfaces to connect with the frame relay providers, and a local fast Ethernet port to connect with the local T LANs, either 100BaseT LAN or 10BaseT LAN. Each HSSI interface is connected

to one of the frame relay clouds. A "primary" frame relay provider is assigned to the first HSSI port, and port speed will be allocated to service the branch. The speed is in the 56K-256K range, depending on the size of the branch. A "secondary" frame relay provider is assigned for the second HSSI port, and has a "zero" port speed connection. This configuration provides a negligible bandwidth circuit in place. Thus, in the case of a primary network failure, traffic will be routed on the secondary circuit, and simultaneously, the network provider can be instructed to raise the port speed on the secondary circuit.

In a preferred implementation of this sample implementation, the table below shows a GEFS data center to remote site WAN configuration.

	Initial	Growth Increment [add 1 DS3 channel pair]
Number of Remote Sites	1200	~680 remote sites
Initial Bandwidth per Site	avg. 128K	avg. 128K
Total Frame Relay Bandwidth into each data center	128K x 1200 nodes = 153Mb/sec	88Mb. per DS3 pair
Configuration [sites distributed evenly across 3 providers]	2 DS3 Channels per Frame Relay provider at 25 Mb, upgradable to 44 Mb in place. (~150Mb total)	1 DS3 Channel per Frame Relay provider at 44 Mb, in place.

Significantly, because the frame relay clouds exist within the private portions of the telecommunications infrastructure, there is a minor vulnerability to intrusion or monitoring. This vulnerability can be completely avoided by insuring that all BEA TUXEDO transactions containing sensitive data are encrypted before transmission over the WAN. Indeed, BEA TUXEDO has state-

of-the-art encryption technology built into its architecture, and this feature can be easily integrated into GEFS system 200.

Furthermore, in addition to the frame relays as depicted by the frame relay clouds shown in Fig. 8, an emerging technology known as virtual private networks (VPN) enables secure point-to-point connections over public networks, such as the Internet. As this technology becomes proven, it will likely become possible to incorporate a VPN cloud as a tier in the GEFS data center to remote site WAN configuration, with the WAN either as a primary or emergency circuit. Using this existing configuration, the architecture of the WAN would allow easy integration of additional tiers employing new transport technology.

As for the internal data center LAN, that is, for the LAN within the data center, the routers are connected to a 100Mb/sec 100BaseT Switched Ethernet concentrator hub system. This hub includes a high-availability, SNMP manageable, Virtual LAN (VLAN) capable system, such as the Cisco Catalyst 5000 Series. The relevant features of this concentrator system include redundant power and cooling, SNMP management, integration with the Cisco Works HA software (enabling failing routers to reassign their nodes to the surviving router), partitioning of Hub ports into VLANS to assist HA and loading balance, extremely high performance switching system, and future slot upgrades to gigabit ethernet, fiber distribute data interface (FDDI), and ATM.

Figure 10 is a schematic representation for the extranet connections, as utilized by the WAN from Fig. 7, and Figure 11 is a schematic representation of the types of clients to be serviced by the extranet connections, as shown in Fig. 10. The preferred implementations are described below.

As shown in Figs. 10-11, the data centers may need to service client applications interfacing remotely over private networks. To take one example, e.g., for banking applications, these client

applications over private networks may include: Internet banking clients, browser-based clearing clients, EDI/Private Banking clients, and remote kiosks and smart cards.

As for internet banking clients, these clients include personal clients participating in "bank from home" programs. Service will be provided as a web service via an internet service provider into an Internet/extranet service center, such as service center 380 in Fig. 3.

As for browser-based clearing clients, these clients include business clients participating in "browser-based-clearing" programs. Service will also be provided as a web service via an Internet service provider into an Internet/extranet service center, such as service center 380 in Fig. 3.

As for EDI/private banking clients, these clients include business clients who wish "branch" type access to their bank. Service will be provided as a virtual private TCP/IP connection via the Internet to a private banking application service at an Internet/extranet service center, such as service center 380 in Fig. 3.

As for remote kiosks and smart cards, these services include Java/TCP-IP based clients that require remote public access to the GEFS services. Typically, an ISP or private network provider will provide a connection from these clients at the point-of-sale or kiosk to the GEFS Internet/extranet service center, such as service center 380 in Fig. 3. A browser-based kiosk can interact with the Web server at the service center, while the smart card can interact with an application server, which services the Java client request.

As shown in Fig. 11, for extranet security considerations, the Internet service bureau, web services, and private connections will be treated as a separate "branch" in the network topology to isolate the back-end GEFS operations from this "virtual front-end" service. This topology is used for enhanced security, although this branch may actually physically reside in the same facilities.

As also shown in Fig. 11, the GEFS network topology provides application access to several external data sources, as well as pass-through to legacy service bureaus, as-needed. In general, the architecture will provide for gateway application machines in the data centers, whose main role will be to (1) provide a physical connection to the external data source via their legacy network; (2) provide some level of data translation from the legacy application to the GEFS data format (e.g., EBCDIC->ASCII, TN3270->ASCII); (3) provide a software interface for transacting data with the remote data source to the BEA TUXEDO/GEFS environment; and (4) act as a gateway or firewall to the legacy network.

In a preferred implementation, the external networks are IBM mainframe centric. This preferred implementation uses standard Sun Microsystems hardware and widely available third-party software as needed to provide these services (i.e., such as FEDWIRE). For example, for banking transactions, the U.S. government mandates that an isolated PC receive data via a proprietary network card and network, where the data is off-loaded manually at the branch, and then uploaded to the bank's database. In GEFS system 200, this activity will be moved and aggregated to the GEFS data centers. Thus, the initial external data sources that will be interfaced with GEFS system 200 include the following: Swiftwire, Fedwire, ACH, ATM networks, and Stock/Press Feeds.

As for internationalization considerations, the adaptation of frame relay and T3 circuits in international locations may require modification of the GEFS network architecture. Nevertheless, I and II network providers provide extensive partnerships and network connections in second-world and some third-world countries. Therefore, for internationalization considerations, selection of a network provider involves evaluation of service offerings in each potential international GEFS deployment country.

b. Processing by Logical Architecture

As this section explains, in a preferred implementation, GEFS system 200 acts as a primary enabling technology for remote application services (JAVA/JOLT) as well as software management (remote software maintenance). In this preferred implementation, GEFS network 205 uses, for example, the JAVA programming language to develop new software programs for processing financial transactions and exchanging financial information over a distributed network. Therefore, users have access to GEFS applications from any type of computer front-end client that supports JAVA. These features off-load the MIS responsibilities at the branch level and allow the GEFS server centers to manage the client applications and network centrally. Some of the preferred implementations are described below.

In GEFS logical architecture 500, applications running on servers update the data in all related tables in a single transaction to ensure that all of them are either updated or not updated atomically. The applications then update data in primary database in real time. Data in remote database might not be updated in real time, however, depending on the criticality of data. Thus, noncritical data in primary and remote databases are not always synchronized, and remote database updates will lag primary database updates. Also, financial auditing and logging capabilities are available to investigate data mismatches. Some of the updates use the XA protocol (e.g., if multiple data managers are to be updated) or native database interfaces based upon specific requirements.

The middleware application uses two-phase commit if more than one database (i.e., resource manager) is to be updated in a transaction. The databases participate in XA protocol if initiated by middleware on behalf of application. The databases also participate in two-phase commit if initiated by middleware.

GEFS network 205 also accommodates the integration of existing legacy components as well as emerging technologies as they prove viable for this application. For example, the application gateway services connect with a gateway router to external legacy networks, including ACH, Swiftwire, Fedwire, ATM networks (including Cirrus, MAC and Plus systems). The fast Ethernet connection also connects with routers that transfer messages between the data centers over the high speed network backbone. The routers also connect with frame relay switches that connect with the frame relay clouds. For banking applications, frame relay clouds provide access to retail banking branches, call centers, and various Java-based application kiosks. The routers also connect with a firewall to provide secure access to Internet users. The Internet provides access through browser-based clearing clients, extranet Internet banking clients, and virtual private access EDI services.

c. Preferred Features of Logical Architecture

The principal financial services applications and functionalities on GEFS system 200 (i.e., the application modules) include, but are not limited to, core retail banking; browser-based clearing (BBC); Global 1000 (GK1); virtual private extranet banking (VPEB); elite private banking (EPB); brokerage and securities; advanced analytics; smart cards; digital credit card processing (DCCP); reinsurance applications; pass-through services; call centers; electronic compliance reporting (ECR); and Y2K solutions. These functions, which represent different application modules for GEFS system 200, are explained in greater detail below.

For the core banking functions, GEFS system 200 entails banking functions in the retail, commercial, and wholesale banking sectors. These core banking functions include such universal financial needs as depository services, loan management, and payment and transfer capabilities.

For BBC, GEFS system 200 incorporates a next-generation digital Fedwire, ACH, and Chips functionality offering encrypted, real-time transfer capacity around the clock, with a comprehensive audit trail and an overall level of security (authentication and validation via Java-based smart cards). This capability includes all forms of trade transactions at borders, seaports, and airports. Further, this capability also includes maritime browser-based clearing (MBBC), which is specifically targeted to real-time, completely secure, letter-of-credit clearance for vessels in port, saving the high costs and fees associated with port "down time."

For GK1, GEFS system 200 allows the world's 1,000 largest corporations to bring many of their financial functions, including corporate treasury management, self-banking, and direct electronic payments to suppliers, inside the corporate structure. A transparent utility permits this in GEFS system 200.

For VPEB, GEFS system 200 integrates the latest in Internet and on-line technology, offering a direct banking solution, housed in a flexible, open architecture, and available and operating in a completely secure and protected environment. VPEB allows tier 2 and tier 3 banks, as well as any financial services provider with a high-value customer base, to have the ability to provide an extranet-based banking and financial transactions infrastructure to their own customers.

For EPB, GEFS system 200 provides functionality in an extended range of specialized, highly customizable, proactive financial services for both high net worth customers and smaller domestic and international money-management groups. Specific EPB functionalities include portfolio monitoring, personalized market updates, virtual personal balance sheets, customized analytics, secure data feeds, and Internet service providership.

For brokerage and securities, GEFS system 200 accommodates real-time global clearing and settling for the purchase and sale of securities through the GFU financial transaction dedicated virtual private extranet.

For advanced analytics, GEFS system 200 makes advanced data and portfolio analytics, as well as information services like Reuters, available on request. In addition, GEFS system 200 provides any data source with a real-time data feed, which can be delivered to the desktop of a user.

For a smart card, GEFS system 200 uses a Java-based, banking-centered smart card with an open, Java-based interface and an encrypted, secure transaction protocol as its central "currency". The Java-based smart card can be used not only for making electronic transactions and purchases, but also for receiving payments, such as salary or benefits payments, through an ATM-like device sitting atop a digital automated data-processing infrastructure.

For DCCP, GEFS system 200 provides streamlined credit-card processing for a full range of credit-card applications, including point-of-sale commerce and Web-based transactions. The DCCP functionality of GEFS system 200 incorporates digital Java-based devices and customized credit-card clearing and processing capabilities.

For reinsurance applications, GEFS system 200 provides a full range of functionality. In particular, GEFS system 200 aggregates data for life insurance, casualty insurance, and other key forms of risk exchange for use by users whose lines of business depend upon having such data.

For "pass-through services," GEFS system 200 provides a variety of financial OEM services, such as life, home, and automobile insurance, by allowing financial institutions to deliver these services within the same overall electronic environment, instead of having to wrap themselves

around a legacy system to deliver such services, especially one that does not operate with their marketing and branch office systems. GEFS system 200 provides these services.

For call centers, GEFS system 200 incorporates a next-generation, secure, and individually brandable call center, usable by all forms of financial services providers and other transactions-based clients, regardless of size. The call center operation is driven by the integrated GEFS database and enhanced by the ability of GEFS system 200 to manage data efficiently by providing a three-tiered call center capability for customer information requests. GEFS system 200 creates customer profiles based on account numbers and direct calls, accordingly.

For ECR, GEFS system 200 provides an electronic compliance reporting system that involves a complete reengineering of the financial regulatory compliance system at federal, state, and local levels, allowing users to reduce their compliance costs dramatically. By recent estimates, such compliance now accounts for an average of thirty percent of total financial services institution earnings.

For Y2K solutions, GEFS system 200 anticipates and remedies electronic problems associated with Y2K.

As demonstrated by the above functionalities and applications, GEFS system 200 integrates and improves the tools already available for migrating data from mainframe-based databases to UNIX relational databases. Indeed, GEFS network 205 can also sustain growth over time without affecting processes that interact with the applications. GEFS logical architecture 500 allows GEFS network 205 to add or change processing power and reconfigure applications to handle changes in load. Indeed, as shown above, the GEFS financial services infrastructure supports multiple financial

services. The range and size of the financial services applications require systems that can support very high volumes of transactions and large amounts of data in a highly available manner.

C. Network Middleware

As shown in Fig. 3, GEFS physical architecture 300 includes application servers 350. These servers allow a variety of clients to "plug-in" to the GEFS utility. The client/server infrastructure that drives this utility is implemented using several components, one of which is middleware (TP Monitors). In a preferred implementation, BEA TUXEDO, from BEA Systems, is used as the middleware for GEFS system 200.

BEA TUXEDO provides a distributed application programming framework that simplifies the development and management of three-tier client/server applications for the enterprise and Internet. The BEA TUXEDO environment hides the heterogeneity of the computers, application programs, and the location of the application programs. As a result, application components can be moved from one location to another easily. It also provides a centralized administration subsystem that allows application administrators to control all nodes that are a part of an environment from one central location. The BEA TUXEDO environment, which is a software infrastructure platform on which to build applications, achieves scalability by an architecture that is modular, flexible, and open; makes efficient use of database, network and system resources; and enables an application development standard that defines coding, security, and new service standards.

With regard to applications, the BEA TUXEDO environment provides for applications that are written as a set of interoperable, modular components using a standard computer language such as C, C++, and Java, or using a high level 4GL tool. These applications servers are then run on BEA TUXEDO's scalable, high-performance, secure, transactional environment. Applications running

on workstation/desktop/webtop nodes are requesters of services offered by application servers running on other computer nodes. A full spectrum of program-to-program communication models are supported, including store-and-forward, asynchronous, RPC, conversational, reliable queuing, unsolicited notification, and event brokering, thereby enabling flexible coupling among the application components. The ability to construct complex enterprise business applications from, modular, well-defined functional components is the forte of BEA TUXEDO. More importantly, BEA TUXEDO enables database and hardware independence.

1. Technical Criteria

Scalability is the ability of an application system software to handle changes, generally incenses, in load and continue to operate within the preset performance and availability limits. In middleware, scalability is central to every large distributed client/server systems application, such as GEFS system 200. Thus, for middleware, a system is required that can use to maximum advantage all heterogenous resources that might be placed on a network for the preferred implementation. BEA TUXEDO provides this functionality through several scalability features.

Vertical scaling is supported by BEA TUXEDO in the sense of migrating or upgrading the system to a bigger, more powerful platform of the same or different architecture. This capability might also include adding incremental processors to a symmetric multi-processing server.

Horizontal scaling is also supported by BEA TUXEDO, which describes a common distributed systems approach and provides for incrementing a group of network-resident processors with additional systems. The aggregate processor group may be homogenous or a collection of heterogeneous processors, such as different microprocessor engines or different operating systems.

"Matrix" scalability is further supported by BEA TUXEDO. Matrix scalability introduces the concept of adding heterogeneous resources any place in the configuration without altering the existing application architecture. Generally, a complex configuration mix can be supported, providing wide-ranging choices for scaling the on-line network system. Issues of data representation (e.g., different microprocessor representations of floating point numbers) are resolved transparently by the BEA TUXEDO system protocols.

In addition to horizontal scaling, vertical scaling and mixed scaling, additional scalability is supported by BEA TUXEDO, providing a logical, three-tier, client-server architecture that is modular, flexible, and open, using database and network system resources efficiently, and creating a GEFS application development standard that defines coding, security, and new service creation standards that enhance modular code development.

Finally, BEA TUXEDO is scalable because of its flexibility in the application environment. As the size of an application grows or becomes more geographically distributed, it becomes inefficient to execute and difficult to administer all application and data components. BEA TUXEDO offers several features that facilitate the building of very large applications. For example, the "domains feature" of BEA TUXEDO allows one BEA TUXEDO application environment to seamlessly interact with another BEA TUXEDO application environment while maintaining separate administration domains. Figure 12 is a schematic representation of the domain structure for a middleware application as shown in Fig. 4. In a preferred implementation, BEA TUXEDO is the middleware application with a domain structure. In BEA TUXEDO, each domain defines an application, or collection of applications, that may span multiple computer nodes but is independently administered. Domains can be distributed across different geographic locations,

across machines, across departments of a company, or even between different companies. Accordingly, the domain gateway architecture of BEA TUXEDO extends the scope of this client/server model to provide for application growth. Significantly, within each domain, the administrator determines which local services will be made available to other specific domains. Therefore, client applications can request services that may be available on another domain.

A single BEA TUXEDO domain can contain between one and 100 machines. Each BEA TUXEDO domain can be connected to more than 100 other domains. Using the BEA TUXEDO's domain capability allows the GEFS infrastructure to accommodate growth.

In addition to the "domain feature," BEA TUXEDO contains other features useful to GEFS system 200. These features include throughput and response time, resource control, processing flexibility, storage flexibility, notification, administration, management, interfacing, migration, and recovery.

For throughput and response time in any middleware application, some of the major factors for achieving the desired throughput are the ability to: use system resources efficiently, ensure availability of these resources, request prioritization, and use the right programming paradigm. BEA TUXEDO achieves high throughput and response time as well as the efficient use of resources by providing both static and dynamic load balancing techniques. In BEA TUXEDO, transaction application server processes can be automatically "brought up" or "shut down" depending on predefined criteria and the dynamic system state. For example, to optimize throughput and response time for a specific critical application service, BEA TUXEDO can replicate the server processes on the same or different nodes, giving more available processing power to service incoming requests.

For resource control, BEA TUXEDO is aware of the state of the resources under its control and uses this information to provide maximum availability for critical applications. Multiple discrete nodes in a distributed system translate into more potential points-of-failure, but also greater potential to redistribute work throughout the remaining nodes while resource recovery is initiated. BEA TUXEDO provides distributed system failure recovery. Typically, in the event of hardware or communication network failures, BEA TUXEDO restarts application processes and provides fail-over to processes on other nodes. Allocating and controlling resources is a key function of BEA TUXEDO. This aspect is even more critical in the distributed computing arena. As computing resources are deployed over networks, BEA TUXEDO can distribute the transaction computation to optimize overall system performance. For example, if one machine is running at capacity, work is off-loaded to another machine at the direction of BEA TUXEDO. Accordingly, the features of BEA TUXEDO request prioritization, request routing, dynamic configuration changes, and different communication paradigms represent an application architecture in BEA TUXEDO that is robust, fast, and meets response time requirements for a distributed system.

For processing flexibility, GEFS system 200 preferably keeps only the most recent (user configuration time) data in the database used for on-line transaction processing. Aged data is moved to a secondary database on a secondary machine. GEFS system 200 uses BEA TUXEDO's data dependent routing feature, which facilitates request routing based on request data. This feature will be used to route incoming requests either to on-line or aged databases transparently to the user. Using all of these facilities allows GEFS system 200 to provide users with a combination of throughput and response for processing flexibility. BEA TUXEDO also allows additional flexibility in processing. For example, some GEFS applications might require that a certain type of message

be given a higher priority over another (e.g., placing brokerage orders has a higher priority than inquiring about the balance in an account) BEA TUXEDO provides a request prioritization scheme to give a class of requests a higher priority, as well as a method of prioritizing a single request from a client.

For storage flexibility, GEFS system 200 might require communications between storage components that are not available at same time. In such cases, GEFS system 200 may need to store requests and process them when the appropriate server component becomes available. BEA TUXEDO supports such batch storage and retrieval of messages through its stored messaging facility, called queue services. Messages are stored in disk-resident message queues and later retrieved by a queue services mechanism. The queue services mechanism supports the model of many real-world business processes. When one step of a business task is finished, the interim results are passed to the next step. The responsible agent for the next task may not be available immediately to process the request, so work collects in a logical in-basket until the next stage is ready to process the request. This is a typical workflow form of software processing. With this queue services mechanism, the message queue is available to receive requests, but actual processing of the request is deferred.

For notification, in GEFS system 200, a user or a program must sometimes be notified of an event to make a decision. For example, a stock broker may want to be notified if a particular stock reaches a certain price, or a performance monitoring application needs to be notified if a server dies, or if a certain performance threshold is reached. BEA TUXEDO provides a transactional event system based on a publish-subscribe programming paradigm, called EventBroker. EventBroker allows for the posting of system or application events, which can be subscribed to by any authorized

client, server, or application component in the system. Depending upon what action a subscriber has specified should take place, the subscriber can be directly notified, a program can be executed, the notification can be placed on a queue for further processing, or the event can simply be logged.

For administration, GEFS system 200 requires middleware where administration has a complex and critical task. In GEFS system 200, resources are distributed, connected by networks, and subject to a variety of conditions over which the administrator does not always have control. For example, the various GEFS financial services modules will offer a variety of financial applications that will be geographically distributed. Managing these applications effectively is critical to customer satisfaction and requires excellent administration. BEA TUXEDO contains features that manage distributed applications. The Web-based graphical administrative utility is an administrative interface into an administrative information base that facilitates the creation of administrative applications. The Web-based administration system provides a secure, Web/Java-based graphical environment. Authentication and encryption capabilities provide an added dimension of security for the Web-based system administrator. Using the administration system, the administrator can closely monitor and manage the distributed application while it is running and optimize its performance or reconfigure the system dynamically. Administrative interfaces include a comprehensive command-line/script interface, a programmatic interface, and a management information base (MIB) for implementing a managed application within a larger administrative framework. The easy-to-use graphical user interface (GUI)-based administration application makes use of the administrative interfaces, providing a single point of high level control over the BEA TUXEDO. In addition, as described earlier, one of the features that BEA TUXEDO provides is the

ability to divide an application into domains. Indeed, GEFS system 200 utilizes different domains to make the applications more manageable.

For management, BEA TUXEDO includes BEA Manager, which helps build SNMP agents that monitor the health of the applications and, where necessary, supplement the system management tools. The BEA Manager interfaces with and feeds SNMP data to a system monitor. The parameters being monitored include the performance and health of applications and databases.

For interfacing, GEFS system 200 interfaces with numerous financial systems and with outside services, such as NACHA, SWIFT, ATM, EDI, etc. To make the system transparent to outside interfaces, GEFS system 200 employs an encapsulation mechanism. GEFS will be the network infrastructure for several financial products and services whose interface requirements cannot be completely anticipated in advance, so GEFS system 200 has open interfaces which will readily accommodate new devices and systems as needed.

For migration, GEFS system 200 contemplates migration from a mainframe in a staged manner. Initially, some components of an application will run on the mainframe. There may be a need for these components to communicate with components running on GEFS system 200. Furthermore, GEFS network 205 must interface with third-party, main frame-based applications. BEA TUXEDO/SNA Domains provides this connectivity for purposes of migration.

For recovery, GEFS system 200 provides data replication between data centers to recover from failure situations. For recovery from database instance outage, GEFS system 200 uses the BEA TUXEDO monitor to check connectivity to the database servers periodically. When a failure is detected, the monitor shuts down all application servers and restarts them so they connect to the respective alternate database server. For recovery from cluster node outage, a database server node

failure will be handled similarly. Also, if the node has BEA TUXEDO application servers, the transaction logs will be migrated to the backup node. Finally, for recovery from a BEA TUXEDO application process outage, if any application server dies, it is restarted by the BEA TUXEDO system server. If any of the BEA TUXEDO system servers die, they will be restarted by other BEA TUXEDO system servers. For in-flight transaction recovery, the client component of GEFS system 200 detects that the connection to a data center is lost and will do the following: if the client is controlling the transaction it will abort the current transaction, reconnect to an alternate data center, and resubmit the transaction; if the transaction is initiated at the server, then the client will reconnect to an alternate data center and submit a test transaction to verify whether the in-flight transaction was successfully committed or not; and if the test transaction returns a failure, indicating that the transaction did not commit successfully, then the client will resubmit the service request, and whenever a database server on the original node is available again, it will reconnect the application servers to the original node and migrate transaction logs, which may be automatic, controlled by the system administrator, or disabled completely.

2. Operation

Figure 13 is a block diagram of an application running on parallel database (PDB) clusters. Fig. 13 presents an example of the operation of BEA TUXEDO, such as, for example, in GEFS system 200. In this example, there is a BEA TUXEDO client (Client Host) requesting a service which is being offered by BEA TUXEDO server A (on node W). Server A connects to RDBMS instance 1 (on Node Y) to satisfy the client request. Node Y is the BEA TUXEDO master and node Z is the backup. Here, there is no failure and the application performs properly.

Figure 14 is a block diagram showing how an application running on PDB clusters recovers from an RDBMS failure. In this example, the same process takes place as in Fig. 13, except here, the client's transaction fails. The steps required to handle this are as follows:

(1) Detect the fail-over condition, shut down the BEA TUXEDO server and start it again so that it gets connected to an alternate database server;

(2) Detect this failure in the client program;

(3) Abort the transaction; and

(4) Resubmit all the requests which were done thus far as part of the current transaction.

As shown in Fig. 14, after the failure, Server A connects to RDBMS instance 2 (on Node Z) to satisfy the client request. Node Z is the BEA TUXEDO backup.

Figure 15 illustrates an additional important feature. This example is similar to that in Fig. 14, except that now, instead of just an RDBMS failure, the node running administrative processes (i.e., master node) experienced an outage. As shown in Fig. 15, upon failure, the administrative processes are migrated to a backup node, and Server A (on node W) has been revectored to the alternate RDBMS instance (*i.e.*, RDBMS instance 2). The application server processes (on node Y) would also be migrated to their backup node (not shown). Significantly, the outage, migration, and recovery have occurred without the knowledge of the client application.

With BEA TUXEDO, data replication is also used by GEFS system 200, and this data replication makes the above-described recoveries possible. BEA TUXEDO provides the necessary features, such as reliable queuing, to perform data replication. GEFS system 200 keeps the data at multiple data centers in sync for another level of high availability. Each data center system will work as the primary system for local clients and as a backup system for remote data center systems and

clients. These systems have several requirements. For example, backup data center systems will be updated as soon as possible after a primary system update. Also, backup data center systems will also be a hot standby, ready to take over as soon as the primary data center system it is down. Specific replication services and durable queues achieve these requirements. To do this, a set of replication servers are running at each data center. These replication servers call the replication services from the fail-over data centers in response to application services called in the first data center by clients. If any data center is not accessible directly for a request to mirror an update, the request to update that backup system will be queued at multiple available data centers. Therefore, the inaccessible data center comes back up, it will apply changes stored in these queues before going back on-line.

3. Operation Scenario

In order to appreciate the functionality of data replication with BEA TUXEDO, the following presents a topical operation scenario. Using BEA TUXEDO, any client in a primary data center can make an application service call, and this call will go to any available server. This server will perform the application logic and will call a replication service to propagate this request to mirror any updates. Replication servers will have two options: 1) call an update service on fail-over data center within caller's transaction, or 2) put this request in a disk based queue within caller's transaction. For critical transactions, replication servers will use option 1, so that the other system can take over immediately in case of failures. For non-critical transaction, replication servers will use option 2 and data will be applied to the remote database as an aftereffect. These types of transactions will not be immediately available on remote machines in case of a failure, but the transaction response time will not be affected substantially because of remote database update.

Figure 16 is a schematic representation of the interaction among replication servers, upon failure of a data center. As shown in Fig. 16, the fail-over data center (data center 2) is inaccessible. If the replication server (on data center 1) is unable to reach a remote system, it will try a configurable number of times to ensure that the system is truly down and not just a slow network/system. If the remote system is not reached, the replication server puts the request in a disk queue for the target system.

Figure 17 is a schematic representation of the interaction among replication servers, upon recovery of a failed data center. As shown in Fig. 17, when a downed system comes back or a partitioned system becomes available again, the recovery consists of the following steps:

(1) Drain the remotely queued requests; and

(2) Switch remote servers from queuing mirror requests to directly calling services.

Following these steps, the replication server again is completely synchronized and available.

To implement the above-described operations, BEA TUXEDO relies on the following security features: authentication; authorization; and encryption.

For authentication BEA TUXEDO implements security by validating a user's credentials. The client captures authentication data and passes it to BEA TUXEDO. BEA TUXEDO then calls an authentication service and allows the connection request to proceed only if the authentication service succeeds. The authentication service involves an industry standard, high strength authentication mechanism.

For authorization, BEA TUXEDO implements security by controlling access to authenticated users. BEA TUXEDO provides access control lists to control access to individual users or groups of users based on services, queues, and events.

For encryption, BEA TUXEDO implements security by providing data encryption capabilities over network links, such as link-level encryption. BEA TUXEDO provides privacy for any transmitted data over the network, which could be accessed by an unauthorized entity. All data transmitted between machines (i.e., between client and server, between server machines, between system and its management console) is encrypted. Encryption can be achieved using either 40-bit (128 bit key with 88 bit disclosure) or 128 bit (128 bit key with no disclosure). The GEFS system 200 decides what level to use based on the security requirements for each individual application and also based on industry standards. Encryption is activated by administrative parameters and negotiation between both parties. Encryption level can be changed at any time using the administrator interface. BEA TUXEDO uses session keys to encrypt data. There are two keys, one for each direction. These keys are randomly generated at network connect time. These keys are agreed upon between two parties using the Diffie-Hellman algorithm, except that DES is used between Java client and GEFS servers. These keys are then used for session duration to encrypt messages. They are discarded at network link drop time.

In addition to the above-described security features, BEA TUXEDO also creates diagnostic and system messages, as appropriate. All output messages are stored in catalogs so they can be easily translated and modified as needed. With this storage feature, date, time and currency representations can also be tailored to conform to the conventions of the user's country. In that regard, BEA TUXEDO conforms to X/Open's Internationalization guidelines as specified in the X/Open Portability Guide (XPG). GEFS system 200 follows the same standards and use catalogs for output messages. Furthermore, GEFS system 200 uses alternative encryption technology compatible with local and international laws.

For diagnostic messages, GEFS system 200 relies on BEA TUXEDO for logging at the service level, but is instrumented to output performance information with service codes and in client codes. Coding guidelines contain information regarding the placement of performance logging messages in code. Logging occurs at the business function level and the data access function level. A post-processor parses the log file and generates required statistics. Furthermore, application instrumentation puts hooks in application code, so that a variety of information can be identified for collection in a running system. In this regard, in a preferred implementation, GEFS system 200 will be extensively instrumented using BEA Manager application program interfaces (APIs). Thus, the application will pass only the information to be logged. Also, BEA Manager will be configured to decide at runtime the type of information logging to be enabled, the log destination, the escalation mechanism and exception triggering.

For error messages, GEFS system 200 divides messages into four categories: error messages, warning messages, informational messages and debug messages. Error messages are not suppressible. Whenever GEFS system 200 encounters any error message, the system returns a text message and an error code to the caller of the service. The system also logs the error message. Under a preferred implementation, BEA Manager is configured to raise a SNMP trigger and/or page the administrator if the error message is serious. The associated parameters are configurable. For example, if an error is encountered while processing a batch, depending on the specific case, the system either aborts the batch on the first error, aborts if more than 'n' errors are encountered, or continues, and reports back that errors were encountered. As for the warning message, the informational message, and the debug message, these other messages (*i.e.*, other than error messages) can be dynamically enabled/disabled in a running system.

D. Network Software

1. GEFS Areas of Operations

Figure 18 is a schematic representation of the three major areas of operations and administration within the GEFS environment. These areas include: system and network monitoring, systems and network administration, and change control.

a. System and Network Monitoring

Figure 19 is a schematic representation showing the monitoring architecture of a GEFS system. As illustrated by Fig. 19, system and network monitoring involves monitoring all systems within the GEFS environment. Monitoring encompasses the process by which system, software, and network components are watched and interrogated to ensure their well being. As shown in Fig. 19, some of the monitoring are monitoring components of GEFS system 200 include data system monitoring, remote site monitoring, and software components monitoring. Data center monitoring includes: database servers, application servers, data warehouse servers, historical data servers, image storage servers, enterprise application (OLAP, compliance) servers, network components, and gateway hardware. Remote site monitoring includes: remote site servers and network components. Software components monitoring includes: BEA TUXEDO, Oracle DBMS, Oracle Parallel Servers, and GEFS application software (e.g., GEFS modules). Two types of monitoring are used for system and network monitoring: SNMP Monitoring and Out of Band/Off Network Monitoring

Simple Network Management Protocol (SNMP) Monitoring is the most common method of monitoring. SNMP is an industry-wide standard that provides a low-level network protocol capable of gathering information on the health of network components and systems. It is widely used in varying types of networks, including WANs and the Internet. All system and network components

are listed as SNMP manageable. This means that the components have an associated SNMP agent. An SNMP agent is a piece of software that is written to answer SNMP requests for information. Even software components such as BEA TUXEDO can have agents associated with them so that their status can be monitored in this manner. Examples of component-specific agents include: Sun Servers and Clients (Sun Domain Manager Agents), Cisco Network Components (CiscoWorks Agents), BEA TUXEDO (BEA Manager), and Oracle SNMP Objects.

In a preferred implementation, GEFS system 200 uses a cooperative console solution, where all SNMP packages report to one monitoring screen. Some of the products that could be used are Solstice Cooperative Consoles, Tivoli Distributed Management Environment, and BMC Patrol. Therefore, in the case of a network or network component failure, connectivity to a managed site may be cut off. In these situations, a backup Out of Band method of reaching the site should be implemented. This can be accomplished through the use of simple terminal server and modem connections to each managed site. Networked terminal servers connected to modems in the Network monitoring center can be used to dial up the branch site or data center currently off the network. The connection on the other side of the phone call is also a modem and a terminal server. The data center or branch terminal server can then be used as a console for any of the systems at the site. The possible security concern over using a terminal server and a modem solution is mitigated through the use of a one-time password solution such as Secure ID or Enigma. These software and hardware packages challenge the user with a randomly generated code, which is then used to generate a short-life one-time only password. This type of password would be required for access to any of the terminal servers.

b. System and Network Administration

System and network administration is provided by middleware applications. As described in the preceding section, in a preferred implementation, BEA TUXEDO is used as the middleware for GEFS system 200.

c. Change Control

GEFS system 200 includes change control software, in a preferred implementation. Change control describes the process of maintaining an accurate record of network connections, network components, servers, systems, and software configurations. The change control database contains information on connections, network components, servers, and workstations, and the type of information includes location, responsibility, user, hardware configurations, software configurations, IP addresses, and connection speeds. The deposit of change control information on a change control database involves the development of a central store of this information and, most importantly, the implementation of rigid guidelines for documentation of the changes.

Change control also generally includes maintaining a record of the applications in a large distributed environment, particularly different revisions of such applications. Controlling the changes of these versions that exist on various server and client platforms is a major challenge. Different platforms will have different possibilities for distribution methods. Push technologies are normally used for Webtop systems or distributed web servers in which objects such as Java applets or data sets are kept locally to increase performance. Also, script-based remote distribution can be used to distribute software packages and configuration files on servers throughout the environment. For these reasons, change control is critical. Generally, the combination of a remote distribution

solution and standard Solaris package installation tools is the basis of a solution for change control for many servers on a network.

In addition to change control for applications, change control is also used for other software. Indeed, new versions of operating system software and updates may need to be installed and controlled. Traditionally, when an administrator receives operating system revisions, this person would be required to physically install this software on each system. However, Solaris Jumpstart is a tool that uses the network booting capabilities of Solaris to boot the system and install the operating system or updates over the network. Jumpstart installations can be tailored to the type of system, disk space, or any other resources of the target. A Jumpstart installation can be initiated from the target system itself or from a remote system with the proper access, by a manual command line process. A server must be configured previously to handle these requests and the software to be distributed must reside on this server. Thus, all of these capabilities fall within the scope of change control.

2. Preferred Features

In this section, a general overview is provided for the network software used by GEFS system 200. Below, the particular software (and hardware) items used in a preferred implementation of GEFS system 200 are recited.

Java/BEA Jolt is a small footprint, non-proprietary client language to facilitate distribution of presentation layer software components to a wide range of client types.

BEA TUXEDO is a robust transaction processing monitor middleware platform to encapsulate application logic as services, manage access to system resources, and guarantee the ACID (atomicity, consistency, isolation, and durability) properties of transactions.

Oracle is a proven, relational database management system to efficiently manage data access in a highly standardized manner.

Sun Solaris is a powerful, multi-threaded operating system to run servers and realize maximum use of hardware resources.

Sun Enterprise class servers are symmetric multi-processing machines for servers to ensure process-level scaling.

Sun Parallel Database (PDB)/Oracle Parallel Database (OPS) is clustering technology to ensure redundancy and peak transaction load performance.

Meshed frame relay WANs, running TCP/IP, from reliable, well established vendors such as AT&T, Genuity, MFS, or Sprint, Cisco routers, and 100baseT LANs in data centers are high bandwidth virtual and dedicated redundant WANs and high bandwidth LANs, both running standardized communication protocols to ensure communications connectivity and performance between tiers.

E. Network Database

1. Overview

Figure 20 is a block diagram showing the data replication system utilized by GEFS system 200. As described in previous sections, the data replication feature is an important aspect of GEFS network 205. Another important consideration of the database in GEFS network 205 is the ability to address future demands in a controlled manner. In GEFS system 200, the location of each new physical database is transparent to the corresponding application. As a result, storage requirements are immense, correlating with the roll out of new applications and the increase in the customer base. Data in GEFS system 200 is preferably sized on the order of terabytes. For example, the Jailbreak

application and its associated databases, as described below, will account for over 1Tb of data. To achieve the required scalability in storage for this and other applications, the data will be split among a set of data centers and a set of databases within those data centers. Having multiple data centers gives protection against disaster whether natural or man-made. After all, as shown in Fig. 20, each data center acts as a backup to one other data center.

For applications on a system as large as GEFS system 200, the database sizes tend to be very large and distributed. The preferred implementation of the GEFS network infrastructure uses the Oracle DBMS as a standard. Still, all components accessing the database can use ANSI SQL, so GEFS 205 network can use any appropriate database based on specific project, hardware, and storage requirements. Nonetheless, Oracle parallel servers (OPS) remain the preferred implementation because it not only allows scaling but also increases capacity, throughput, response time, and system availability and reliability.

For capacity, Oracle OPS provides the ability to support large volumes of data, thousands of users and high volumes of batch, and on-line transactions. This ability includes the capability of sharing resources among many requesters while ensuring that response times and throughputs do not suffer during peak workloads. Also, the Oracle allows for distribution of data by horizontal partitioning. This capacity allows for practically infinite data capacity while maintaining required performance.

For throughput, Oracle OPS satisfies the requirements for the sum total of all transactions across all application modules and machines that GEFS system 200 is capable of handling in a given time period. In one implementation the typical throughput of GEFS system 200 is expected to be around thirty billion (2 billion/module x 15 application modules) transactions per year. A fully

configured Sun Enterprise 10000 system is capable of handling 1348 transactions/second. In a clustered configuration, with up to sixteen machines/cluster, the hardware can be scaled as needed. Accordingly, thousands of clients can access GEFS network 205 concurrently, so the application architecture and the applications themselves will be distributed and built in a manner to meet the required throughput.

For response time, Oracle OPS is capable of meeting the perceived time it takes the system to respond to user input. In one implementation, thousands of users will depend on high performance from GEFS System 200 using the Sun Enterprise 10000; the response time depends on several elements, including the networking infrastructure, database design, client and application server architecture, and operations management. A well-designed online system provides a response time that serves the business needs of the business it is designed for. GEFS network 205 ensures that response times are met by specifying the performance requirements for all critical transactions and by conducting performance tests.

For system availability and reliability, Oracle OPS supports a myriad of network solutions, where each network configuration provides the reliable and secure information network required. The use of BEA TUXEDO will provide prioritization, request routing, dynamic configuration changes, event handling, asynchronous communication and other such features that are used to create a robust and fast application architecture capable of ensuring both availability and reliability.

2. **Preferred Architecture**

Figure 21 is a block diagram showing a detailed view of a data center for the network database. Data Center 2100 includes BEA TUXEDO servers 2110 connected by communications link 2115 to databases 2120. Significantly, in a preferred implementation, each database 2120

includes multiple Oracle parallel servers (OPS). The communications link also connects with communication gateway servers 2130 that provide access to and manage the routers (not shown).

The architecture of a network database, as shown in Fig. 21, provides for significant operating and administrative savings. The thin client workstations provide central administration, management, and distribution of new client software, which provide substantial cost savings. Furthermore, an intuitive interface within Java-based clients minimizes end user training and support costs. Thus new financial applications and services can be downloaded to each thin client as needed, thereby efficiently integrating with existing client applications. This capability of GEFS 205 dramatically reduces the cost to install and implement new business applications.

The architecture of a network database, as shown in Fig. 21, also provides for high availability. The redundant and high availability design of GEFS network 205 enables efficient decision support and online analytical processing (OLAP) functionality. The redundant information repository and the ability of users to interconnect provide for customer marketing and tracking report capabilities.

The architecture of a network database, as shown in Fig. 21, also supports knowledge management capabilities. For example, using the flexible Oracle Relational Data Base (ORDB), connectivity to existing legacy sources and inherent redundancy allows GEFS network 205 to provide a cost effective platform for financial knowledge management; GEFS System 200 provides relevant financial information through data mining, data warehousing, decision support, OLAP, and ad hoc queries. Thus GEFS system 200 provides the ability to understand and market financial services effectively to targeted groups of customers.

Using the architecture of a network database, as shown in Fig. 21, the data must be routed to the appropriate multiple data centers (and databases). As described above, using the TUXEDO middleware, GEFS system 200 contains software technology that facilitates seamless routing to the correct multiple data centers (and databases). Thus, applications do not need to be embedded with specific instructions indicating where the data will be stored. This middleware technology makes it easier to develop applications, a technology capability referred to as data transparency or partitioning.

For data transparency (or partitioning), each database stores data from a range of sources (e.g., 1000 different banks). The data will be stored in such a way as to achieve independence for each of the entities, e.g., one bank will not be able to access the records of another bank. Therefore, even though the data is stored in one database, the data itself is independent. This is referred to as a logical partitioning and the purpose of this capability is to allow the application elements to link to and interact with other elements. Traditionally, this has not been possible due to the disparate nature of the application (e.g., a banking system has no knowledge of any insurance policies that the customer may have). Under a customer-centric database system, however, all the accounts can be linked under the one customer record.

In order to provide this customer-centric database system, GEFS system 200 uses a single provider of database technology for reasons of interoperability. There are significant advantages in this approach: for instance, streamlined change control and centralized management. However, another key advantage is the ability to link the databases together. Thus, using single provider, database links are easy to maintain and provide the ability to swap data seamlessly without the worry of data interchange formats. Due to this customer-centric data system, in a preferred

implementation, GEFS system 200 utilizes a concept of billing the GEFS customer (an insurance company, for example) on a transaction basis. The customer will have a range of services available to it and will have the option to select one service or many, with billing based on the number of transactions for any or all services. Thus, the customer is only charged for actual use of a service.

Indeed, a standard requirement in any data system is the need to query the data for aggregated information. Most systems do not allow these queries to be run during the day because they seriously impact the transaction work and degrade performance. Yet, whenever allowed, there are two types of queries: (1) the ad-hoc query and (2) the on-line analytical processing (OLAP) query. In an ad-hoc query, a user runs a singular, non-repeated query. Some transaction processing (TP) systems allow ad-hoc queries to be run against the live database, but the majority do not. In an OLAP query, data is required from many sources. In this type of query, therefore, the demands are very intensive in terms of CPU and disk resources. Therefore, OLAP processing is always done on a separate machine. This secondary machine is often referred to as a data warehouse. The data is usually extracted periodically (daily, weekly, etc.) for the data warehouse with no impact on the live system. GEFS system 200 provides for both types of queries, i.e., both ad-hoc queries and OLAP queries. OLAP queries against the range of databases on GEFS system 200 could deliver unprecedented, high value, marketing information to GEFS customers. However, the data warehouse for all the GEFS applications could exceed 50 terabytes in size. Due to this possibility, the data redundance of the network database in GEFS system 200 is significant.

One of the keys of GEFS system 200 is the availability of the data in the network database. To attain the highest availability to the data, the GEFS system 200 must be able to deal with a total data center failure. To do this, data is stored in an alternate location to its primary site. Using the

features of BEA TUXEDO such as queuing (BEA TUXEDO/Q) and the distributed transaction interface (XA), data is inserted into the secondary database using reliable BEA TUXEDO capabilities. This secondary database will be housed in the n+1 data center. For example, as shown in Fig 20, if the primary records are stored in data center two, then the backup copy will be in data center three. The last data center will backup its copy of the data to data center one as also shown in Fig 20.

As with the primary copy of the data in the primary database in the primary data center, the secondary copy of the data is stored in another database in another data center. This database is a fully functional copy that can be used to perform ad-hoc queries and can be configured to be the primary data source in a matter of seconds, in the event of primary failure. In a common implementation, a flag stored in the database indicates whether it is currently the primary or secondary source.

In GEFS system 200, availability of data is also ensured for the most recent data. In current systems, the most recent data is the most commonly accessed data. For this reason, customers are frequently concerned with items that have occurred within the past couple of months. This is reflected in the database systems where a least recently used (LRU) mechanism is used to keep the most accessed record in the machines memory. However, GEFS system 200 stores the most recently processed data on the fastest devices. One database system will hold live records for the past four months, and a separate application will periodically move these records to a secondary or historical database.

Furthermore, GEFS system 200 also provides for availability of historical data. Each data center in GEFS system 200 includes a historical database, which is a fully functional on-line system

with interactive access. If an application needs historical data, the application does not need to know the target database because the application middleware will route the request to the correct database depending upon the data. In BEA TUXEDO, this feature is known as data dependent routing (DDR). In a preferred implementation, the historical database contains records that are older than four months and younger than two years. Once the record is older than two years, the data is moved to a slower tertiary storage device. The standard media for the tertiary storage device involves optical technology.

3. **Preferred Features**

Figure 22 is a block diagram showing a preferred implementation of a database with two access machines. As shown in Figure 22, in this proferred implementation, an Oracle Parallel Server (OPS) is used to allow two different machines to access a single database. Oracle has two variants of its core 7.3 RDBMS product, either single instance or OPS. The single instance version is the most common. In the single instance version, one machine (or node) accesses its own database and all requests for the data come through that single instance. This variant gives greater fault tolerance from a failure of an individual element, e.g., hardware, operating system, application or database instance. Thus, if any one element fails, then the alternate node can continue operations on the database. However, GEFS system 200 used OPS for even greater advantages.

Indeed, as previously described, GEFS system 200 uses the latest parallel technologies available from Sun and the Oracle RDBMS for the database network, which provides support for parallel hardware technologies through the OPS software.

Using this OPS Software, in the event of a failure, BEA TUXEDO automatically re-routes work through an alternative Oracle instance. This switching of Oracle instances is achieved in a

matter of seconds and results in an minimal delay in processing the application software's request. Furthermore, no transactions are lost, because the alternate Oracle instances recover the transaction of any failed instances.

Use of the OPS Software only involves one possible problem, namely, if two Oracle instances try to access the same data block. However, in a preferred implementation of GEFS system 200, the problem is resolved through the use of a distributed lock manager (DLM). The DLM coordinates data block access to prevent conflicting operations. In particular, the DLM prevents two Oracle instances from updating the same records on two different nodes. If the first node has updated a block and the second node attempts to update the same block, the DLM will force the first node to write out the update in order for the second node to read in the new value. Thus, DLM serializes the transactions to provide data integrity. Further, performance is maintained by ensuring that two nodes operate on different blocks of data. This technique is known as application partitioning and is enforced by data dependent routing in BEA TUXEDO.

For processing of DLM in OPS software, there are two options: dba locking (also known as fine grain locking) and hash locking. Fine grain locking is suitable for systems that have no control over application partitioning. However, GEFS system 200 has been designed with specific routing criteria that promote strict application partitioning over nodes. Therefore, in a preferred implementation, hash locking has been chosen for this environment.

Figure 23 is a schematic representation of conflicting application requests for access to data from the same block, and how such requests are intercepted by the DLM. As shown in Fig. 23, the availability of a database is not just the consideration of failure handling. It is also the consideration of day-to-day activities such as backing up the database, because these functions should not interfere

with the availability of the database. The Oracle software allows a database to be backed up while full production work is being performed. The Oracle term for this method is hot backup, and in a preferred implementation, the databases in GEFS system 200 use this method for ensuring availability as shown in Fig. 23.

Other than the DLM issue, the use of the OPS software in the network database for GEFS system 200 entails the same considerations as any other large database network. However, in the preferred implementation, Oracle software is used for the databases. Using this software, some significant features are realized, which are particularly utilized by GEFS system 200. For example, Oracle uses partitioning, where large tables can be logically divided into smaller partitions. These partitions can be brought on-line and off-line without affecting the rest of the table, hence making maintenance a very easy process. Also, Oracle supports data redundancy. In most transaction processing systems today, database networks can never be taken off-line to bring the software up to date, hence the software is usually vastly out of its supported date range. GEFS system 200 ensures that all system software (e.g., operating system, application, database, etc.) can be upgraded at predetermined points. This can be achieved due to the redundancy of data; using data redundancy, the front-end clients are not directly connected to the database, so they can be routed to an alternate database via the middleware. Thus, if the client is dynamically routed to the alternate database, the primary database can be upgraded and patched.

As described in the preceding sections, data redundancy provides significant protection against failures. Indeed, failure scenarios have to be considered, and one of the keys to the success of any database network is its ability to deal with a failure from any single component. In a preferred implementation, using Oracle, GEFS system 200 has software and hardware architecture to prevent

the following types of failures dealing with the database network: (1) an Oracle instance failure and (2) a total database failure.

Figure 24 is a schematic representation of the recovery steps taken by a database network upon an instance failure.

As shown in Fig. 24, in the event of a single instance failure, such as an Oracle instance failure, the middleware (e.g., BEA TUXEDO) automatically shuts down the send connected to the instance and then restarts pointing to the other Oracle instance. In a manner similar to Fig. 24, in the event of total database failure, all processing will automatically be switched to the alternate data center residing in another physical location. This would be functionally equivalent to the data center outage, as shown in Fig. 14.

In addition to the features described above, with regard to Oracle, GEFS system 200 also relies on two additional features: the Enterprise Manager and the Simple Network Management Protocol. Enterprise Manager (EM), is a system management tool that gives the administrator of an oracle database an intuitive, easy-to-use interface. EM runs from a central location and all operations are controlled from the Enterprise Manager Console EMC). From this console, an administrator can administer, diagnose, and tune multiple databases; distribute software to multiple servers and clients, schedule jobs on multiple databases at varying time-intervals, monitor events throughout the network, and run integrated third-party applications and tools. The benefits to the administrator are a centralized console for single point of management, scalability for growing distributed environments, automated ("lights out") system administration, intelligent autonomous agents, and security management. Thus, EM allows a high degree of automation through remote task execution and reactive and proactive management capabilities. EM also scales to accommodate very large

database configurations, while providing sufficient flexiblity to manage installations with numerous databases. Due to the large number of Oracle databases used in a preferred implementation of GEFS system 200, EM plays an important role.

As for Simple Network Management Protocol (SNMP), this is provided by an "agent" (or SNMP agent) in Oracle release 7.3. Oracle SNMP support allows all Oracle7 products to be queried by any SNMP-based network management system such as Sun's Solstice or Tivoli's TME products. Like the EM product, Oracle's SNMP integration allows: (1) Oracle products to be located, identified, and monitored in real time across enterprise networks of any size; (2) database administrators to see the current status of Oracle products, selecting from among several status variables that are defined for each product in a management information base (MIB); (3) monitoring of the status of Oracle services such as databases or tools throughout the network; (4) identification of performance bottlenecks; (5) "discovery" of Oracle databases or tools as they start up on any node; (6) alert generation when exceptional events occur, for instance, the database going down; (7) definition of thresholds and automatic responses to specific events; (8) rapid detection and diagnosis of potential problems; and (9) storing, reporting upon, filtering, and analysis of historical data. Like EM, GEFS system 200 also utilizes SNMP.

Finally, as described previously, Oracle provides for logical storage of data on a database to avoid contention between the data of various users. This takes place by logical partioning. In addition to logical partitioning, the database network also provides for physical partitioning to preclude instance contention between processes and nodes. For example, if datafiles 1-10 are assigned to instance one, and datafiles 11-20 are assigned to a second instance, BEA TUXEDO routes the database commands to the appropriate instance. Thus, for a more specific example, users

(or customers) with a last name in the range a-m would be routed to instance 1 and users (or customers) n-z would be routed to instance 2. With the combination of logical partioning, physical partitioning, and middleware routing, contention will be minimized and the database network will have the performance characteristics of two separate databases, which is the desired scalability result. In particular, by using the Oracle disk affinity/allocation feature, no instance contention would be observed. This Oracle feature avoids the OPS pitfall of "pinging," where two Oracle instances try to access records in the same operating system block. Upon the release of Oracle 8, these partitioning issues will be implemented even more securely. Oracle 8 will take the partitioning feature further with data partitioning. In general, partitioning addresses the key problem of supporting very large tables and indexes by allowing users to decompose them into smaller and more manageable pieces, called partitions. Partitioning can significantly reduce the impact of scheduled downtime for maintenance operations. This greater efficiency occurs in two ways: (1) by introducing partition maintenance operations that operate on an individual partition rather than on an entire table or index; and (2) by providing partition independence, so that maintenance operations can be performed concurrently on different partitions. Thus, whenever a record is inserted into the primary database, it is essential that the backup copy also contains the record. This backup security is achieved by using the XA interface to enable a two-phase commit approach to data storage. The backup security of the data can be considered to be separated into the following areas: Oracle database connection; Oracle data; application; and database sizing. Oracle is a fully XA-compliant resource manager and integrates seamlessly with the BEA TUXEDO system. Thus, if direct access is not available, BEA TUXEDO allows access to Oracle software.

Figure 25 is a schematic representation showing a database network connection to the database through middleware, such as BEA TUXEDO. As shown in Figure 25, because of the lack of a direct connection to the database software, access to the database is available only through the application middleware (e.g., BEA TUXEDO). Therefore, the connection string to the database is only known by the application. However, this means that even if users had an Oracle application on a PC, they would not be able to connect to the database because they would not possess the connect string information. When a user logs into an application, the access level is obtained for that user. For this reason, the functions that a user can perform through the Middleware are dictated by this access level. In BEA TUXDEO, this delineation of allowable functions is implemented by Access Control Lists (ACL).

4. Redundancy

Figure 26 is a schematic representation of data center redundance for a WAN. As shown in Fig. 26, data center 2610, data center 2620 and all other data centers 2630 connect to all of the frame relays 2640, 2650, 2660. Each frame relay 2640, 2650, 2660 is preferably provided by a different service provider. For example, frame relay 2640 may be a Sprint device, and frame relay 2650 may be an AT&T device, or any other virtual private network. VPN 2660 may use the Internet. As further shown in Fig. 26, each call center 2610, 2620, 2630 routes messages to at least two frame relay providers. Thus, even if one frame relay goes down, the redundant frame relays can still route messages from the call centers and bank branches. Furthermore, if one data center goes down, the other data centers will still be online. As shown in Fig. 26, in a preferred implementation, data centers 2610, 2610, 2630 include: reliable and redundant communication lines (preferably dedicated); an uninterruptible power supply (UPS) at each data center to protect against power loss;

a redundant air-conditioning system; and proper building selection and preparation, which protects against natural disasters, such as fire, flood, earthquake and the like. Furthermore, to provide additional security, at least two data centers are located in two different geographic locations and are operating in load sharing mode, as shown in Fig. 26. When one of the systems cannot operate because of some unforeseen event, all clients of this data center will migrate to surviving data centers until repairs or corrections can be made.

Figure 27 is a schematic representation of the redundancy of communication links among multiple data centers. Data center 2610, data center 2620 and additional data centers 2630 are connected by multiple T3 or ATM service provides, such as, for example, AT&T, MFS, or Sprint. Therefore, with the redundance of connections between data centers, even in the event of network wide failure, connectivity between data centers is insured.

Figure 28 is a schematic representation of two centers with the normal operational flow of transactions through frame relay clouds. As shown in Fig. 28, eastern branch bank clients 2810 typically route messages through a frame relay cloud 2820 to east data center 2830. Likewise, western branch bank clients 2840 route their transactions through the frame relay cloud 2820 to the west data center 2850. East data center 2830 and west data center 2850 each contain a mirror of the other's database as well as its own live database. The mirrors of each database are updated on the fly and continuously.

Figure 29 is a schematic representation of two data centers, where the flow of transactions through frame relay clouds has been interrupted by a data center failure. As shown in Fig. 29, if east data center 2830, for example, goes down, then eastern branch bank clients 2810 will be routed to east mirror database 2860 in west data center 2850 and the transaction server will process eastern

branch bank clients 2810 through the newly-live east mirror database 2860, which was a mirror of the live east database 2870 before east data center 2830 went down. When normal operations are restored, the newly-live east database 2870 is updated with the data from the west data center east mirror database 2860.

Thus, to implement the redudancies described in Figs. 26-29, in a preferred implementation, a Sun high availability platform is used, which has two or more machines operating concurrently, each with a hot standby for the other. For this redudancy, this system disk arrays are mirrored and accessible from all nodes of each cluster. To detect faults, the platform provides capabilities to detect database failures and reconnect the server to a backup database in the cluster; detect machine/network failure/partition and mark it as down; repair faults automatically if possible or, if not, raise events for administrator manual intervention; or update data in a database of the remote location. Further, the platform allows for updates of the data as needed by critical transaction types in real time so that the other system can take over immediately in case of failures. For noncritical transactions, the platform creates a log on the remote machine and applies data to the database after the effect. These types of transactions will not be immediately available on remote machines in case of failure, but delaying the updates for noncritical transactions will not affect transaction response time too much. Accordingly, through the use of such redundancies, a network database of this type provides the maintenance of continuous client information. Whenever a client receives a brief indication of failure or time-out, the client will quickly determine that the failure or time-out is because of a localized network or machine and not because of a centralized network or machine problem. Such a database network is embodied by a preferred implementation in GEFS system 200,

F. Clients and Outside Systems

1. Client Computing

The client computing architecture in GEFS system 200 is based on an open, extensible, and flexible architecture that allows for the execution of transactions on a wide range of client platforms. The guiding principle of the GEFS client computing architecture is to construct an environment that enables GEFS system 200 to provide seamless, secure and cost-effective transactions across a wide range of devices and communication channels.

a. Overview

As shown in Fig. 2B, GEFS system 200 anticipates three categories of clients: on-time clients 225, batch clients 227, and DSS clients 229. However, in a preferred implementation there are three specific types of clients envisioned for interacting with GEFS system 200: internal transaction clients; gateway clients; and OLAP clients. Internal transaction clients are part of GEFS infrastructure, such as customer service representatives, tellers, etc. These clients are clients for the purposes of BEA TUXEDO and directly submit transactions on behalf of users. These include Java Station configurations. Gateway clients provide interfaces to external devices and systems. These processes work as protocol converters and submit TUXEDO transactions on behalf of external systems and also submit external requests on behalf of GEFS applications. OLAP clients execute dynamic, ad hoc database queries and these queries might return unpredictable (and most of the time large) amounts of data from database. These processes go directly to the database, avoiding middleware. Because these processes degrade system performance, they are used judiciously. In a preferred implementation, GEFS network 205 uses the concept of a thin client as often as possible. In particular, network computers are the client platform of choice. These machines support the Java OS as its operating system. The interface utilizes the icon-based interface within the Java Station.

b. Interfaces

i. Clients

For interfacing clients and outside systems with GEFS network 205, a variety of interfaces are used. For clients, a network interface is most likely.

In a preferred implementation, GEFS network 205 is designed to exploit the "HotJava Views" by Sun on the desktop. This provides access to E-mail, calendar, and names in a directory in addition to access to the user interface. This interface is only for the Sun Network Computers. For other computers, existing PC-based workstations and Internet clients using a browser-based interface are also used to interface GEFS network 205. One alternative to installing JavaStations or other network computers is to enable existing workstations under the context of a Java-enabled browser client software as provided by Netscape Navigator® or Microsoft Explorer®. Any Java-enabled browser enables GEFS network 205 to support existing hardware and software configurations inexpensively.

ii. Outside Systems

For outside systems, there are several means of interfacing with GEFS network 205. Direct dial-up clients may connect to GEFS network 205 through a modem bank for example, with home banking. For these clients, the GEFS network 205 provides a secure way of modem connectivity. GEFS network 205 is also accessible using Web browsers. Indeed, because the client component of applications is developed in Java, access from the Internet is automatic. Further, the security component of the Java system will handle issues related to Internet security. Finally, an interface is also possible via financial systems and interfaces, such as NACHA, SWIFT, ATM, and EDI. GEFS network 205 provides good encapsulation to keep most of these outside systems transparent of outside interfaces. Yet, because GEFS network 205 contains a technical infrastructure with

several financial products and services, open interfaces are available, so the network can easily interface with new devices and systems needed by new applications and systems.

iii. Other Interfacing

As for other interfacing with GEFS network 205, network computers and PCs are typically used as the interface by transaction workers (e.g., tellers, customer service representatives, loan approval and processing agents, etc.). However, GEFS network 205 also supports other types of interfacing by client devices as needed such as Voice Response Units (VRUs), Magnetic Stripe Readers (MSRs), call center systems, and hand-held wireless units.

VRUs provide automated phone support.

MSRs are credit/debit/ATM card readers at stores and banks.

Call center systems interact with or are part of call routing by PBX. They get data about the caller, such as name, account number, and account history, from back-end systems and provide that data to service representatives.

Hand-held wireless units can be Personal Digital Assistants (PDAs), such as a Sharp Wizard®, Apple Newton®, or 3com PalmPilot®, with built-in wireless modems, and can be used to access GEFS network 2.5.

In addition to the use of client devices for interfacing, gateway processes can also act as pseudoclients and handle transaction I/O from these devices so existing interfaces of these devices do not have to be changed. There are also daemon processes that process external input batch file, submit transactions, and optionally create output response file for external use. For example, GEFS network 205 provides gateways to interfaces with external systems, such as NACHA (National Automated Clearing House Association); ATM/Store POS devices, such as Star/Explorer, Maestro,

Interlink; Interbank wire transfer network (SWIFT network); EDI credit card authorization networks, such as VISA, MasterCard, Discover, American Express, Diners Club, JBS; call center systems (e.g., if a bank has issued any credit cards of the above mentioned type, authorization requests will come to the issuer bank from the respective network); check services networks such as Telecheck, Telecredit, JBS, Mercantile; image processing system for images of signatures, photos, collateral items; and check/statement/passbook printing machines. GEFS network 205 also provides file level interface with popular financial packages like Microsoft Money® and Intuit Quicken®. GEFS network 205 outputs files that these packages can understand, and allows extraction of data from files created by these packages. GEFS network 205 also provides interface to auxiliary devices like receipt/passbook/checkbook printers.

A Java Computing Architecture is used for the GEFS client computing architecture. In a preferred implementation, the GEFS client computing architecture supports the following two general classes of clients: Java-based and Java-enabled clients and traditional clients, with existing external system interfaces, batch interfaces, and server-centric processes.

As for Java-based and Java-enabled clients, the GEFS Java computing architecture is a key component of the overall GEFS client computing architecture. The GEFS Java computing architecture supports both Java-based and Java-enabled clients. A Java-based client is a GEFS-supported client platform that executes a Java operating system as its native operating system. Examples of Java-based clients include Sun JavaStations and Java-based smart cards. A Java-enabled client is a GEFS-supported client platform, based on a non-Java operating system that contains a Java Virtual Machine. Examples of Java-enabled clients include windows 32-bit based PC clients. Nevertheless, whether a Java-based or Java-enabled client, the GEFS Java computing

architecture provides for a high degree of connectivity and commonality among the various software components. Most existing, new, and emerging GEFS clients will be based on a GEFS Java computing architecture. This commonality is achieved by GEFS leveraging of Java's "Write Once, Run Anywhere" paradigm, which enables the reuse of GEFS application modules on multiple GEFS-certified client platforms.

As for traditional clients, the GEFS client computing architecture is also able to support transactions and interactions with traditional systems. For financial transactions, examples of these systems are interfaces to systems such as NACHA, ATM, SWIFT. Many of the systems are based on non-real-time information exchange. In such cases, the GEFS client computing architecture supports transactional information interchange in batch formats with these traditional external systems.

iv. Technical Criteria

If a client satisfies the GEFS client computing, then the client will be supported for purposes of the GEFS client computing architecture. The GEFS technical criteria are satisfied if the following criteria are met regarding scalability, capacity, throughput, response time, manageability, accessibility, availability, interoperability, useability, ease of development, robustness, and data integrity and security.

For scalability, the GEFS technical criteria prefer that growth of new clients in the client tier be accommodated by the inherent scalability in the middle and back end tier servers.

For capacity, the GEFS technical criteria prefer that, where appropriate, clients, particularly batch, DSS, and OLAP clients, will be certified to work with large amounts of data.

For throughput, the GEFS technical criteria prefer that clients be certified to provide adequate throughput for all applications that can execute on their respective client platforms.

For response time, the GEFS technical criteria prefer that response time for clients is ensured by a high speed private network, and use of prioritization services.

For manageability, the GEFS technical criteria prefer that the GEFS system 200 promotes zero administration clients for all client platforms. Further, in cases where zero administration clients are possible, remote manageability features will be utilized.

For accessibility, the GEFS technical criteria prefer that GEFS client computing architecture be based on industry and universally accepted standards, such as IP networks and Web technologies.

For availability, the GEFS technical criteria prefer that Java be supported for multithreading.

For interoperability, the GEFS technical criteria prefer that GEFS client computing architecture leverage Java's "Write Once, Run Anywhere" paradigm to transparently support multiple client platforms.

For usability, the GEFS technical criteria prefer that clients provide universally accepted and easily usable user interfaces such as point-and-click Webtops and touch screens.

For ease of development, the GEFS technical criteria prefer that GEFS system 200 utilize a component development model that promotes the reuse of components amongst various GEFS applications.

For robustness, the GEFS technical criteria prefer that clients will be able to switch transparently to secondary GEFS services in cases of unavailability of their primary GEFS services.

For data integrity and security, clients will participate in a security model to ensure secure and authenticated transactions originating from a certified client platform. For this purpose, in a preferred implementation, GEFS Java clients utilize BEA Jolt.

v. Categories of Client Computing

As shown in Figure 2a, GEFS system 200 includes three primary client categories: online, batch, and decision support system (DSS) clients. These categories of clients (or users) have unique characteristics that impose different requirements on GEFS system 200. The following describes the preferred implementation.

For on-line clients, GEFS system 200 uses Java-enabled programs using the BEA Jolt API for BEA TUXEDO running under a browser. This approach resolves the software manageability issue of client machines. BEA Jolt will be used to provide connectivity to BEA TUXEDO from applets. After booting, these clients download the client application from a Web Server and then follow the described steps.

For batch clients GEFS system 200 uses UNIX processes. Batch clients process data that generally comes from external sources and is to be applied in the GEFS system 200. Batch clients create output files for external systems or run reports. The volume of requests made by batch clients is large, and they impose a significant load on the system. They generally interact with GEFS system 200 during off-peak hours. These client programs are sophisticated and have built-in failure recovery processes and management functions.

For DSS clients, GEFS system 200 uses programs similar to on-line clients. DSS clients will either be applets running under browsers or DSS tools. DSS clients support "canned" queries that

access the transaction server and execute the "canned" queries. These queries are BEA TUXEDO services and allow the client to specify certain display and selection criteria.

Figure 30 is a flow diagram showing the interaction between a client and the GEFS system. Specifically, Fig. 30 illustrates the interaction between a client and GEFS system 200 in accordance with a three-tier client/server application request. Notably, the steps taken do not depend on whether the client is a batch client, online client , or DSS client. As shown in Fig. 30, after a client boots (step 3005), it requests to join a managed GEFS application managed by BEA TUXEDO (e.g., retail banking) and supplies authentication and authorization data as well as the name of the communication server process to which it wants to connect (step 3010). A session is established with the logical communication server process, and the client-supplied authentication data is encrypted and sent over the network (step 3015). The communication server receives the "join" request and passes the supplied authentication/authorization data to a security server (step 3020). The security server determines the client's authenticity and passes the information back to the communication server (step 3025). Based on the security server's response, the communication server then either allows or rejects the client's "join" request (step 3030). Once the client has joined the application, it calls a BEA TUXEDO-managed service (e.g., GEFS business service, Deposit) (step 3035). A request message is then formatted and passed over the network to the communication server (step 3040). This message can be encrypted. The communication server interrogates the BEA TUXEDO Bulletin Board (BB) name server to select a transaction server process (advertising the appropriate service) to which to route this request (step 3045). The communication server also examines the request data and performs any necessary routing based on the data content (step 3050). The request message is then passed from the communication server process directly to the selected

server for processing (step 3055). The transaction server process receiving the request invokes the requested named service (step 3060). Typically, a service will call a data server that then accesses a resource manager, perhaps performing an update. The data server might be collocated on the same node as the transaction server or, alternatively, might be executing on another remote server. Depending on the request type or content, the transaction server may queue the request on the queue server for staged processing (step 3065).

2. Java Computing

Java-based technologies, though relatively new in the computing industry, have been successfully developed and deployed in many mission critical systems. Java is now the preferred language for the development of financial services systems at many of the world's largest financial institutions. Although the primary guiding force of Java was initially Sun MicroSystems, all of the major technology industry leaders have endorsed and released products based on Java computing. These companies, most of which have committed and directed large investments towards Java-based technologies, include IBM, Oracle, Microsoft, Cisco, and Netscape.

Like the other components of the GEFS client computing architecture, the GEFS Java computing Architecture is designed to address robustness, security, and data integrity as well as software life cycle issues such as revision control, maintenance, and deployment. In the three-tier GEFS client computing architecture, the middle tier servers provide distributed resource provisioning, data access, and client support by leveraging BEA middleware and Intranet technologies. In the GEFS client computing architecture the top tier inherits and builds upon the secure, robust networked environment that is provided by the other tiers. This client tier capability

enables GEFS to provide lightweight, intelligent, and secure financial services applications on a wide range of Java-based and Java-enabled clients.

a. Overview

The major benefits of the GEFS Java client computing architecture include secure and authenticated services; portable, compact, and platform independent applications; an intuitive and easy to use graphical user interface (GUI) for applications; zero-administration clients for most supported client platforms; reliable client platforms; and internationalization support.

i. Technical Criteria

The GEFS Java client computing architecture highly leverages Java's unique "Write Once, Run Anywhere" capabilities. Java is platform-independent, thus allowing the same GEFS financial services applications to run on all types of client platforms, such as network computers (NCs), Personal Computers (PCs), UNIX systems, Personal Digital Assistants (PDAs), kiosks, set-top boxes, smart cards, and smart appliances. The platform independence features of Java will be leveraged by the GEFS Java client computing architecture to support a wide range of Java-based and Java-enabled devices cost effectively. These features of the GEFS Java client computing architecture enable GEFS to certify, support, and brand both existing and new Java-based and Java-enabled platforms with its unique "Runs on GEFS" mark.

The GEFS Java client computing architecture uniquely leverages the Java computing paradigm in conjunction with BEA middleware products and a robust security infrastructure to allow GEFS services to be transacted securely from traditional, new, and emerging access devices and communication channels. This inherent capability of GEFS will provide a significant competitive advantage over existing systems by seamlessly providing secure access across a wide range of

existing and emerging communication devices and channels. Existing systems must usually develop additional systems and modules to support new devices and channels. GEFS Java applications run securely on most existing access devices and are likely to be supported on all emerging access devices. A robust GEFS security infrastructure based on market-leading technologies, such as BEA TUXEDO security, ensures that client platforms certified as a branded "Runs on GEFS" client will contain the financial services industry's leading security and authentication features. The GEFS Java computing architecture provides an authentication model in which users will be authenticated against BEA TUXEDO security services.

GEFS Java applications contain the many interfacing advantages of Java-based and Java-enabled clients. Indeed, GEFS Java applications that use a GUI are designed to consider human engineering factors, aesthetics, and intuitiveness of use for the target client platform and intended audience. For example, GEFS Java applications used by financial services industry workers in a corporate intranet environment predominantly utilize an intuitive GEFS Webtop based GUI. The GEFS Webtop is a web browser-like GUI that enables a financial services industry worker to point and click to icons on the GEFS GUI to access GEFS applications. The GEFS Webtop interface is similar to the Webtop interface implemented in Sun's HotJava Views® product. The use of such a GEFS Webtop by corporate users will substantially increase productivity relative to current host-centric dumb terminal interfaces, or even Windows-based interfaces.

GEFS Webtop GUIs allow for easy client interfacing with GEFS Java applications. Plus, the universal acceptance of Web technologies by the general public will enable the easy assimilation of GEFS products and services into the consumer marketplace. GEFS services targeted to the consumer marketplace will follow the GEFS principle of providing an easy to use GUI on each

GEFS-certified client platform. For example, the GUI for GEFS kiosk-based services will likely contain a touch screen interface. In addition, the use of GEFS applications will significantly reduce the required training time and costs that are typically required for traditional corporate GUI interfaces. This greatly reduced training burden is a significant advantage to GEFS corporate customers, especially due to the transient employment cycles of many financial services industry workers.

All GEFS-certified Java-based clients will be zero administration clients.

Additionally every effort will be made to deploy GEFS Java-enabled clients as fixed function devices which are zero or near zero administration clients. A zero administration client is a significant economic benefit of the GEFS client computing architecture. Various studies have estimated a five-to-one cost savings advantage of network-centric zero administration clients versus traditional PCs.

Nonetheless, The GEFS client computing architecture supports certified Java-enabled clients, such as PCs, when required. When supporting the PC platform, the GEFS client computing architecture attempts to reduce the high administration and management costs associated with the PC platform. Reducing administration and maintenance costs for GEFS Java enabled devices such as PCs is accomplished by providing a GEFS client computing environment that provides for on-demand downloading of GEFS applications from centralized servers to the Webtop-based PCs. However, the costs to support certain non-GEFS preferred devices, such as PCs in a corporate environment, will still be significantly higher than GEFS zero administration clients, such as JavaStations.

The GEFS client computing architecture allows GEFS users to interact with GEFS in their natural language and according to their customs. GEFS Java applications can be developed so that they tailor themselves to the user's language and customs. In particular, Java supports display of: UNICODE characters; a locale mechanism; localized message support; locale-sensitive date, time, time zone, and number handling; collation services; character set converters; and parameter formatting. Furthermore, due to the flexibility provided by Java-based and Java-enabled clients., transactions may also be leveraged by TUXEDO's robust middleware transaction capabilities. In fact, due in part to Java, most GEFS clients will be able to detect a failed connection to their primary data center, and seamlessly establish a connection to their secondary data center.

ii. Architectural Principles

The GEFS Java client computing architecture is based on the principle of small segments of capability and component models. In general, most GEFS Java applications will be designed to provide small modules of functionality. This will allow the applications to load onto the client platform quickly, since it may be possible to load only certain classes of a GEFS Java application initially, and then load other functionality ondemand or at a later specified time during the application's execution. Although there are other criteria involved when applying this principle, a general adherence to this principle will allow for the reasonable performance of GEFS Java Applications on other Java-based and enabled devices such as PDAs, set-top boxes, and smart cards. These Java-enabled devices will be much more resource constrained than NCs and PCs and may not be operating in a networked environment. Therefore, it is imperative to design the GEFS Java applications with considerations of the various client platforms on which they may be deployed.

Based on this component model principle, the GEFS Java client computing architecture allows for the definition and creation of software components which can be dynamically combined together to create GEFS Java Applications. Because of the GEFS Java client computing architecture, GEFS applications support the existing and traditional financial services clients such as NACHA, SWIFT, ATM networks, POS networks, check service networks, credit card networks, and credit bureau inquiries and reporting. Support for many of these existing systems and traditional financial services clients will be batch-based processes. In these cases TUXEDO prioritization features will be utilized to ensure adequate throughput and performance for these processes as well as concurrent on-line transactions.

The primary component model for the GEFS Java client computing architecture is JavaBeans, which builds and enhances upon the Java platform. The GEFS Java client computing architecture can utilize the JavaBeans object model to design and build independent software components that can be used and reused to compose distinct GEFS applications and services. JavaBeans is a fully distributed software component model and platform neutral API for creating and using dynamic Java components. JavaBeans can facilitate the design and creation of well-defined component interfaces. This capability is a key factor in allowing for specific functionality to be leveraged amongst present GEFS applications, and future GEFS applications.

G. Security

1. Requirements

Security is a hybrid function of business processes and requires a technical framework to support them. For example, financial environments require security of financial data, while insuring integrity and accessibility to the data. Security must integrate with the end-user environment. In the

financial environment, for example, the end-user environment includes teller stations at bank branches, home or private banking clients, sophisticated financial services, and extranet users from external networks such as the Internet. The GEFS security architecture realizes these dichotomies.

The GEFS security architecture is designed to simplify and centralize security functions for the end-user clients. The security architecture facilitates the development of new applications by providing a consistent security model. The GEFS security model is capable of expanding its scope to several thousand distinct remote clients each with dozens of remote offices. This is possible under the GEFS security model because the security functions of the GEFS security architecture do not interfere with the performance of the client applications. Also, the GEFS system greatly reduces local support by offloading management information system (MIS) functions, such as user management, to the GEFS data centers. Therefore, an effective central user management system for the remote-sites is important for the GEFS security architecture.

2. Security Architecture Components

Figure 31 is a schematic representation showing the hierarchy of the security architecture of GEFS system 200. As shown in Figure 31, the GEFS security architecture directly addresses security at several layers. The GEFS security architecture is specifically targeted at creating several nested layers of security, each protecting the layers underneath. The GEFS security architecture protects financial transactions and financial information by encrypting the actual data between end points using 56-bit private key encryption. The authority of the end-users and machines is insured using private key NIS+ point-to-point authentication, which confirms the identity of the user and originating machine. TUXEDO HA also protects the transactions by insuring that the actual specifics of the transaction are permitted for both the user and application submitting the data. The

GEFS security architecture is also extensible to future security technologies such as X.509. As shown in Figure 31, the hierarchy of the GEFS security architecture includes: machine security, network security, application security, authentication security, physical security, and enterprise security.

a. Machine Security

As shown in Figure 31, the inner-most layer of security in the GEFS security architecture is machine security. Machine security involves the security of the actual hardware at each site. Machine security includes: workstation security, server security, remote site security, data center security, and other system security.

For workstation security, the GEFS security architecture prefers that individual workstation machines at the remote sites are physically isolated to areas of the facility not accessible to the public. The workstations are also locked-out at the firmware level to prevent reconfiguration. This protection will limit the usage of the machines to GEFS applications. Thus, any reconfiguration will require GEFS MIS authentication.

For server security, the GEFS security architecture prefers that the servers are completely locked out at the firmware level. The servers are maintained over the network. Thus, local access will be provided to the server only as necessary.

For remote site security, the GEFS security architecture prefers that GEFS system 200 assumes all administrative and security functions. Thus, local intervention or maintenance will be required, as these functions will be performed via the GEFS data center. This scheme allows the remote sites to be completely locked-down in terms of the GEFS physical infrastructure.

For data center security, the GEFS security architecture prefers that the actual machinery at the data centers be secured in a limited-access area with all network and computer hardware in guarded and/or alarmed areas. Access will only be given to authorized and screened personnel. Monitoring will track and record all entrance and exits of people and materials from the data center. For example, the servers in the data centers will be audited by security software on a regular basis. Checksums (MD5) and other inventory checks will be made on the systems to insure application and data images have not been tampered with. Similar checks will also be performed on other systems (e.g., remote systems).

For other system security, the GEFS system architecture prefers that all hardware be locked at the firmware level, similar to the remote machines, to prevent tampering. Also, the GEFS system architecture prefers that the operating systems be configured to facilitate security in several areas. For example, no extraneous user accounts or files are to be maintained. Also, the central servers will not participate in NIS+ to keep access strictly limited to on-site personnel. Furthermore, all extraneous TCP/IP port services will be turned off at the kernel level, and all relevant activity logging for the user and processes will be turned on.

b. Network Security

As shown in Figure 31, the next layer of security in the GEFS security architecture after machine security is network security. Like machine security, network security involves a layered approach to the security of each component of the network, including: network routing, network auditing, data center WAN to data center WAN connections, data center to remote WAN connections, middleware, Internet connections, extranet connections, external services, and future needs evaluations.

For network routing, all routers that gateway network traffic into the remote sites and data centers will employ hardware port filtering to limit all but the predetermined-as-used services ports on all portions of the network. This filtering will prevent all network traffic of an unauthorized type in or out of any location. In addition, all servers will have all unessential services configured out of their services offering. This will prevent any server from offering services beyond those deemed essential.

For network auditing, particular network security auditing tools, such as COPS, will be run on a regular basis to probe and report on the overall status of open and closed ports. Rules-based monitoring will also be employed to monitor proactively for suspicious events, such as repeated failed entry attempts, sequential probing of ports, use of ports at unexpected hours, unusual system load patterns, and multiple location logins of the same user.

For data center WAN to data center to WAN connections, security issues are minimal. A data center WAN to data center WAN connection is a point-to-point private line (T3) network. The circuits used for these connections are private and dedicated to this service. In general, these lines are safe. Regardless, all transactional data sent via these lines will be encrypted.

For data center WAN to remote data center WAN connections, security issues are also minimal. A data center WAN to remote data center WAN connection uses frame-relay technology to connect remote offices to the data center over a private switched (T3) network, like a data center WAN to data center WAN connection. In a preferred implementation, the network is privately routed within the telecommunications network of a l provider similar to private phone calls. Moreover, the GEFS security architecture provides additional protection by the physical isolation

of the FR network. The guideline for machines at data centers is to strip the machines of extraneous user services, and limit access to and from these machines..

For middleware, all financial content will be encoded by the BEA TUXEDO transaction system, which uses a 56-bit private key encryption between the client application and the TUXEDO servers at the data center. Thus, any data on the frame relay wire between the remote office and the GEFS data centers is at an encryption level equal to or better than current financial applications. In addition, all user and host (remote computer) authentications passed between the remote site and the data center are performed using state of the art NIS+ naming service, which also employs 56-bit private key encryption.

For Internet connections, the GEFS remote service center will be a physically distinct web and application service connected to the Internet. As a result, extensive security measures are implemented. Thus, secure Web-server and Web-client technology, such as RSA and SSL, insure a clear-channel connection between the remote web browser. At the remote service center, users will log-in and authenticate themselves to the remote service center, and then subsequent transactions will be assembled as GEFS transactions to be submitted via a firewall to the GEFS private network, as shown in Figure 3.

For extranet connections, the most extensive security measures in the GEFS security infrastructure are used because these connections interface with open public networks, such as the Internet. Like Internet connections, the GEFS remote service center will act as a secure proxy agent to enter the GEFS system. By dividing the GEFS system from the extranet clients, conventional technologies can be used to insure transactional security, and the other layers of the GEFS security architecture can be used to secure the transactions using the internal systems.

For external services, external datafeeds and application gateway services will typically reside in the data centers with one network interface on the legacy network and the other on the internal GEFS data center LAN. Because these systems could be used to passthrough an outsider into the core GEFS data center, these systems must be evaluated on a case-by-case basis to insure safe operation. Fortunately, most of these systems are connected over private lines to either service bureaus or the government and are not at less risk then being on a public network. Generally, remote clients and remote applications are not directly connected to the GEFS infrastructure. Instead, transactions are assembled by the remote service center application server and passed on. If this is the case, inherent flaws in the client technology are isolated to the client-to-remote connection, and not GEFS network 205. Thus, by focusing on protecting the transaction, the GEFS security architecture may use the proxy host at the remote service center as a transactional firewall, and then add the physical isolation of a TCP/IP firewall at the service center to isolate access to GEFS network 205. This strategy insures the internal transaction network is pure and safe, and offloads authentication and security to the remote service site.

For future needs evaluations, the GEFS security architecture contemplates that, if a security design evaluation deems that additional security is needed at the network level, site-to-site networking can be enhanced with additional hardware (e.g., to provide full DES encryption across the WANs using virtual private tunneling). Also, if security requirements are particularly high, smart card technology can be employed with a user-authentication scheme to provide an additional "authority" service for user logins. Many other options are also available to increase security.

c. Application Security

As shown in Figure 31, the next layer of security in the GEFS security architecture after network security is application security. In general, most of the application security is derived directly from NIS+, TUXEDO, and Oracle. GEFS applications are typically a hybrid of Java Client, Oracle DBMS, and TUXEDO TP interface libraries. As a result, the GEFS security architecture uses the security capabilities of these programs for GEFS applications. In particular, the GEFS security architecture places scopes of control around these subsystems to provide layers of security for GEFS applications. BEA TUXEDO provides one such layer of security.

Because BEA TUXEDO is the middleware in the preferred implementation, BEA TUXEDO is used to protect transactions at various stages of every transactional process. For example, BEA TUXEDO establishes TCP/IP socket connection to the transaction server and immediately negotiates a private key for the life of the transaction. After the key has been established, all subsequent transactions are encrypted across the network. Using this security process, every BEA TUXEDO application service very specifically configures which clients are permitted to transact what types of data to it. In BEA TUXEDO, these scopes of control are called domains. When a client establishes a transactional socket to a TUXEDO server, the server examines every transaction and verifies whether the originating application and client are permitted to request services from the server. This security is a layer of transactional protection above the NIS+ layer, which provides another layer of security in the GEFS security architecture.

BEA TUXEDO is also used for authentication of application security, as compared with general authentication, described below. For application authentication, an application running at the branch (typically Java-based) conducts authentication by the following steps: (1) the TCP/IP client authenticates itself as a node to the NIS+ service with the NIS+ server; (2) the application

opens a socket to the BEA TUXEDO server and negotiates a private key for encrypting the remainder of the transaction; (3) the application obtains and submits the user's name/password to the BEA TUXEDO server; (4) the BEA TUXEDO server queries and authenticates the user's credentials with the NIS+ server; (5) the application assembles transactions and submits them (encrypted) to the BEA TUXEDO server; and (6) the BEA TUXEDO server examines and validates the domain of the user/application/transaction and submits the transaction for processing.

d. Authentication Security

As shown in Figure 31, the next layer of security in the GEFS security architecture after application security is authentication security. The GEFS security architecture authenticates machines and users using a proven and reliable method of network authentication, namely employing NIS+ naming services to manage user accounts and to authenticate individual workstation nodes in the environment. As explained above, NIS+ is also integrated with the TUXEDO authentication services to provide user authorization at the application-level.

NIS+ is a widely accepted system of centralized host and user account authority that provides a secure naming service for TCP/IP environments. It allows a centralized "Master" directory server to aggregate and server lookup services for all authenticated clients to use for login account and hostname-to-address lookup. It employs a system of 56-bit private keys to add machine terminals to the environment initially. All subsequent client-server transactions are also authenticated. Users in the environment are also authenticated networkwide in this manner.

NIS+ employs a system using an authoritative "primary" node and user database that publishes to a set of many secondary or "client" machines. In a preferred implementation, GEFS data centers will provide a central primary node, which will serve highly available "secondary" nodes

at remote sites. NIS+ directly addresses the problems of centralized management via its primary/secondary server architecture. It also addresses the threats of IP and domain spoofing by the NIS+ mechanism of authenticating exact hardware client hosts on the network via its credential mechanism. Furthermore, NIS+ provides a mechanism to add or remove users and hosts from participating in the GEFS environment.

NIS+ is also highly adaptable to all aspects of the GEFS security architecture. To take one example, NIS+ can be migrated to smart cards and X.509 technology, as these and other technologies emerge. These emergent technologies are built on similar architectures of client-server key exchange and centralized database services. In particular, in a smart-card enabled environment, a user would swipe the smart card, enter the PIN code, and then proceed with the normal NIS+ login procedure. This use of a smart card would thus provide an additional tier of authentication for user services. Implemented as a feature of the GEFS security architecture, the use of a smart card would include benefits to scalability, capacity, throughput, manageability, robustness, and internationalization.

For scalability, the distributed model of NIS+ is engineered to centralize control functions of many remote client nodes. Additional sites and users would be added in a centralized fashion.

For capacity, there is no practical limit to the network system size managed by the NIS+ system. The network system can be split into business domains as the network system grows.

For throughput, NIS+ security is an integral component of each software component, so it has negligible performance impact. For response time, NIS+ security is an integral component of each software component, so it has negligible performance impact.

For manageability, NIS+ centralizes management. The architecture maintains basic similar building blocks (i.e., sites/users), which makes adding new objects consistent and straightforward. Key management with NIS+ requires initial setup, but works transparently once installed on a client.

For robustness, NIS+ provides for primary and secondary servers, as well as failover capabilities. For data integrity, NIS+ key authentication insures the identity of end-to-end users and machines.

For internationalization, the modular design of the GEFS security architecture, which includes NIS+, and BEA TUXEDO, allows for the straightforward integration of secure encryption components to comply with export laws. NIS+ provides consistent programming interfaces for other subsystems.

e. Physical Security

As shown in Figure 31, the next, outer layer of security in the GEFS security architecture after authentication security is physical security. Physical security includes the security of the actual machines (e.g., security from intrusion) and facilities (e.g., security from break-in).

f. Enterprise Security

As shown in Figure 31, the outer-most layer of security in the GEFS security architecture after physical security is enterprise security (or meta-security). Enterprise security involves the security of GEFS system 200. For example, the GEFS data centers are to be supported around the clock by on-site system administration and security personnel. In addition, a secondary system of technical and administrative support should be built into the data center support system, and centralized monitoring of all data centers and remote sites insures the physical security at all sites. Furthermore, the remote sites, such as branches, are inherently less controllable than the main data

centers, and the scope of access at these remote sites is specifically restricted to insure enterprise security.

H. Application Module: Jailbreak

As shown in Figure 2A, a number of applications may be "plugged into" the GEFS infrastructure. These applications are called "application modules," or simply "modules." One application module of the overall GEFS concept is called "Jailbreak," which refers to a core retail banking functionality. This application module is called "Jailbreak," because the application module allows banks to "break" free from the "jail" of proprietary network systems for implementing core retail banking functions.

1. Technical Architecture

Jailbreak is a distributed, client/server, scalable, highly available, retail banking system targeted at 2 and 3 banks in the United States. In a preferred implementation, Jailbreak services 1000 banks, but the application architecture allows scaling upwards as necessary. Jailbreak also offers a sustained lower cost of ownership than existing core data systems for retail banking. Plus, the Jailbreak thin-client, no-administration workstation offers a much more secure transaction entry environment. The Jailbreak system also provides interfaces to a range of proprietary external devices and systems, such as ATM networks.

Jailbreak offers many advantages over current systems. For example, Jailbreak is adaptable to new functional requirements. Indeed, the products offered by banks are numerous, and new products are added frequently. Jailbreak is designed around a rules-based engine, so that when banks offer new products, the application can be readily extended to service the new products. For example, a significant operational cost component for a bank is regulatory compliance. Regulations

are numerous; they change frequently. New regulations are added even more frequently. Jailbreak would be expandable to include an electronic compliance application for integration into Jailbreak. Such an e-compliance application for Jailbreak would incorporate a series of manageable components, facilitating the ongoing incorporation of any and all regulatory changes.

Jailbreak is also advantageous because the application accepts data from present systems. The data in existing banks may be placed in two categories. The first category contains those banks that use financial services providers and are tied to a legacy system. These legacy services providers store a bank's information in one mainframe in a remote location. The second category contains those banks that have moved away from legacy services providers to a self-contained client server environment. This self contained client server environment has been provided for them by a technology provider, such as the Open Solutions Inc, Phoenix, and M&I Eastpoint. For banks to be moved from these categories to Jailbreak, the data must be migrated into a relational data format for the Jailbreak Oracle database. The migration of data requires data extraction and transformation tools (such as those commonly used in data warehousing), knowledge of both the source and target data schema, and a well-defined migration process model. Jailbreak functions seamlessly with such migrations and is compatible with all migration methodologies and tools. Tools such as ETI Extract or InfoRefiner from Platinum software perform these tasks.

2. Logical and Physical Overview

As described above, in a preferred implementation, Jailbreak leverages the three-tier distributed computing features of GEFS system 200 by enabling service to at least 1000 banks. To do so, Jailbreak utilizes BEA TUXEDO application services for the banking functions, such as loan products, deposit account products, operations support, report processing, and general ledger. In

addition, both on-line and batch processing will be supported. Furthermore, the use of BEA TUXEDO by Jailbreak provides a rule-based infrastructure, which easily allows for extensibility and changes. Thus, the functions available in Jailbreak are modular and designed in such a manner that interchange of modules is easily accomplished.

The functions of Jailbreak are controlled by an application architecture comprising three-tier, client-server applications. The Jailbreak applications are initially distributed between the customer locations and two data centers. Branches of a typical bank have LAN connecting machines running Java-enabled browsers. These machines download applets from a Web server running at one data center and connect to a communications server running at the same data center. Whenever that data center is unavailable to a client, Jailbreak automatically reconnects to the fail-over data center and continues operating. The exact configurations depend on each bank. For example, a large bank may have a distributed server configuration running at the bank facility instead of within a data center.

3. Hardware

Figure 32 is a schematic representation of the Jailbreak application module showing the architecture of the server infrastructure. As shown in Figure 32, Jailbreak uses the GEFS infrastructure with the addition of a branch server. The branch server is a multipurpose server resident at the bank branch site. The requirements for the branch server are detailed in the following table.

Local Web Server	These systems will serve as a local caching web server for the branch Webtop systems.
Local Jumpstart Server	These systems will serve as network boot/install servers described in section IV.D. "GEFS Areas of Operation."

Local Diskless Webtop Boot Server	Any diskless Webtop system such as a Java station will require a boot server to provide operating system download.
Communications Gateway to Data Center	All data transfer and communication with the assigned data center will be made through these systems.
Local Application Server	Applications that need to be run at the branch site will be served from this system.
Local Network Information, Domain Name, and Key Server	Information such as passwords, host names, and public and private keys will be stored on these systems.

4. **Database**

Figure 33 is a schematic representation of a customer-centric data model, as used by the Jailbreak application modules. The data model applied in the Jailbreak architecture is a customer-centric data model, as opposed to an account-centric data model. In a traditional legacy banking system, a customer may have many different accounts such as checking, savings, credit card, etc. In such a legacy banking system, these accounts are not related to each other in any way, causing wasteful and inconsistent duplication of data. Furthermore, due to the lack of connectivity between the accounts, duplication of effort is often made, such as when a checking account run is made to offer credit cards even though the customer has an existing credit card with the bank. Under the customer-centric data model, shown in Figure 33, all the accounts are linked under the one customer record. This data model thus eliminates duplication of effort and allows a bank to better manage its relationship with its customer. Indeed, because duplication is eliminated, the bank can maintain a comprehensive picture of their customers' activity.

Banking transaction records are generally in the order of 100 bytes in length. Most banks generate 10,000 transactions a day, which implies that 1 Mb of new data is generated by each institution per day. In a preferred implementation, Jailbreak is based upon the storage of data from 1,000 banks hence 1 Gb per day. Over a two year period, this translates to 710 Gb of data. The base records (e.g., name, address, etc.) will also account for another 40-50Gb, and the indexing of the records may be account for another 500Gb of data. Therefore, the total size of the storage requirements for Jailbreak is in the region of 1.5 terabytes. In one implementation, GEFS system 200 will hold two years of on-line data with older records being archived to an archive database server. This archive server will also be on-line and is a part of the application logic to route the query to the appropriate database server. Thus, based upon the assumption that the Jailbreak database would hold over 500Gb of data with a similar amount held in storage for indexes, current database technologies satisfy this implementation, which provides for the storage and management of a database in the size range of up to 2-4Tb. Nevertheless, Jailbreak splits the databases for each bank over a range of redundant databases. Therefore, while the individual records of a particular bank will be contained within one node of one database, to ensure performance for update and retrieval, the database has redundant copies to guarantee ready access.

In Jailbreak, there are two data centers that provide services for each Jailbreak customer. These two data centers provide back up for each other. Additional data centers can be utilized, if capacity requires. As with any application on GEFS system 200, transactions are sent to a respective data center dependent upon the routing criteria in the middleware layer. Yet, data is stored separately in the data centers. The customer-centric data model used by Jailbreak allows the records from banks to be isolated to prevent corruption and security issues from arising. Each record in a bank's

database contains fields indicating the bank and the customer number in order to prevent any one bank from accessing another banks data.

Jailbreak architecture is designed to take full advantage of both the transaction processing and the data warehousing features of Oracle. Features delivered in Oracle 7.3 include bitmap indices, star queries, and enhanced parallel query support for clustered technologies. These capabilities systems allow for standard and advanced reporting facilities for every bank, and the bank can choose which reports to use. The cost is reflected in the number and complexity of the reports chosen. The reporting system can include the necessary documentation required by state laws. Presently, certain banks cannot provide these reports due to the cost and complexity involved in developing the necessary computer systems. This is especially true in 3 banks, where the only option is to out-source the request to a third party provider at a great cost. Jailbreak provides this functionality.

Using the features of Oracle, Jailbreak also supports both types of Decision Support System (DSS) queries. The first type of DSS query is the ad-hoc query, which may be generated within an individual bank. Ad-hoc queries are submitted through a TUXEDO service whereby Oracle will estimate the statistics for execution time, number of rows retrieved, and CPU usage. If any of these values exceed preset parameters, then the query will not be executed. When this situation occurs, the customer can be given the option to schedule the query after normal business hours, modify the query, or simply delete the request. Generally, however, these ad-hoc queries will not generate significant amounts of data and they will only be concerned with data related to a specific bank. In most situations, these queries can be run against a backup copy of the database to prevent any impact on the live production system. The second type of DSS query is the large scale, data

warehouse/OLAP request. These type of queries involve large amounts of data and must be run against an entirely different database. Data warehouses are characterized as aggregations of multiple disparate data sources, which often include inputs which are outside the scope of the core business, such as demographic/government data. Data warehousing queries are run on a separate server. In a preferred implementation, this server is a single-purpose, single machine that would reside in one data center.

Use of Oracle also provides certain special options. For example, with the Oracle Video option, Oracle allows the storage of binary large objects (BLOBS) without the use of an additional option. However, this type of information is not stored alongside standard transaction records. Instead, an index field in the transaction record will refer to the actual location of the multimedia image. This capability may prove useful for various optical or video uses. For example, many banks in the marketplace today allow customers the option of having their photograph embossed onto their credit cards. A more comprehensive list of multimedia functionality includes imaged checks and bank statements, audio, video, and still photographs including fingerprint images.

Upon the release of Oracle 8, even more functionality will be realized by Jailbreak. Just as one example, Oracle 8 provides for full data partitioning, which will allow for greater space management control. This will allow database tables, such as for service and maintenance, to be modified online without disturbing existing connected users.

5. Networks

As shown in the following table, GEFS network 205 acts as the enabling technology for the Jailbreak application module. Jailbreak must have reliable connectivity to the following nodes.

Client Service Request	Server
Installing Operating System on boot	local boot service on branch LAN
Loading of application toolbar	local web service on branch LAN
Loading of Jailbreak applications	local web service on branch LAN
Application transaction	BEA TUXEDO application server at regional datacenter
Name-to-IP address resolution	DNS server on branch LAN
GEFS database services	via BEA TUXEDO application server at regional datacenter
Security/Login service	local NIS+ server on branch LAN

According to the Jailbreak network architecture, most of the bandwidth intensive operations are placed on the local LAN, leaving only maintenance functions and database transactions to travel on the WAN. These services are placed on the local 10BaseT LAN to offset the potential load on the 128K connection to the datacenter. The local bandwidth is several orders of magnitude faster than the WAN, and will ensure excellent application performance. If an increased number of users or applications at a branch create greater database traffic over the WAN, the standard 128K WAN circuit is capable of upgrading to 1.44Mb/sec via a configuration change. If a very large single branch exceeds this bandwidth, multiple circuits can be added. For the premium or larger branches, the use of 100BaseT Ethernet (100Mb/sec) provides a network typically 10 times faster than most legacy 10BaseT (10Mb/sec) networks currently deployed. An exception to this architecture involves Mini-Branches, which are aggregated and served from the GEFS data center on centralized services (mini-branch issues are discussed below).

The Jailbreak network architecture consists of several identical building blocks to aid management and simplify deployment. As shown in the following table, common maintenance activities are performed seamlessly by the Jailbreak network architecture.

Activities	Remote Site Action	Datacenter Action
Installing new Webtop	• Notify GEFS of Ethernet address of new machine • Unbox machine and plug into LAN and "boot net."	Configure central NIS and boot servers to load appropriate images and install media for machine.
Add new user	Notify GEFS of employee name and permission via phone/fax.	Add login to centralized NIS and security server.
Install/upgrade software	Reboot machine.	Configure central publishing server to deliver new SW to branches.
Add new software	Reboot machine.	Configure central publishing server to deliver new SW to branches.
Backups	Nothing.	GEFS backs up all data at datacenter.

Thus, due to this architecture, deployment of a new remote office is made simple and manageable. The specifics of putting an office on the network can be performed remotely and even accomplished from the GEFS operations center. All of the Jailbreak network building blocks are deliberately consistent and manageable by industry standard interfaces, such as SNMP and Java Management Application Program Interface (JMAPI). This network architecture enables GEFS to exploit state-of-the-art network management tools to monitor and control the entire GEFS enterprise from central points of control. The architecture also simplifies deployment and growth of remote offices.

In the Jailbreak network architecture, the remote offices contain SNMP and JMAPI-compliant hardware and software to enable a centralized monitoring system. The centralized GEFS

operations management center has automated monitoring maintenance functionality to monitor and control several functions at the local branch. Any machine outage will automatically trigger an alarm at the operations center. Also, any outage of web service, boot service, DNS, NIS, or security ports will trigger an alarm. Furthermore, any unusual demands or slowdowns will trigger corrective actions, and any failed backup or software upgrade will trigger corrective actions.

According to the Jailbreak network architecture, at the branch level, the network contains several Java-enabled Webtop platforms and a departmental server. Reasonable performance and especially costs are a consideration at the branch. The departmental server at the local branch is intended to provide local network directory and application services. It will ask the local network proxy server for login, security, and network lookups. It will also act as a local java/Web application server. Placing these services locally on the high-speed LAN increases performance for the end-user and reduces bandwidth cost and requirements. The secondary providers for these services will be servers at the GEFS data center, in the event of primary failure. The Jailbreak network architecture uses NIS+ security for access and usage. However, physical security at a remote branch is less controllable than the data center. It is of course essential that there be no intrusion into the security, application, and network databases contained on the departmental server at the branch. For this reason, in a preferred implementation, the departmental server will be completely secured and no access will be provided to the local personnel. The machine will be case-locked, firmware locked, and physically locked at the 05 level. Only remote GEFS datacenter personnel will have access to the machine, and all services provided by the server will be installed, managed, monitored, and controlled by the GEFS datacenter. In the event of failure of the departmental server, services are

automatically failed-over to the GEFS data center servers, which act as the primary server until the local server is repaired or swapped out.

a. Mini-Branch Considerations

A mini-branch is considered to be a small remote office with one to three machines. For a mini-branch, a 10Mb/sec network is connected via two frame-relay channels in the range of 128K to the GEFS data center, which acts as the departmental server. For a mini-branch, the departmental server role is performed over a WAN from the GEFS data center. This "data center departmental server" performs all the tasks normally served by the "branch server." For this reason, additional bandwidth is used for application services. Indeed, client performance must be carefully evaluated during the development and test period for mini-branches as this configuration directly relies on the network for application performance. Notably, in one implementation, this mini-branch configuration depends upon the JavaStation's ability to boot from a local FlashROM card. If this functionality is not available, in a preferred implementation this configuration will require the addition of a local departmental server at the mini-branch.

b. Standard Branch Considerations

A standard branch is considered to be a medium sized remote office with three to ten machines. For a standard branch, a 10Mb/sec network is connected via two frame-relay channels in the range of 56-128KB to the GEFS data center. A local application server supplies all standard GEFS services for the standard branch.

c. Premium Branch Considerations

A premium branch is considered to be a larger remote office with more than ten machines. For a premium branch, a 100Mb/sec network is connected via two frame-relay channels in the range of 128K-256K to the GEFS data center. Like the standard branch, a local application server supplies all standard GEFS services for the premium branch.

6. Client

a. Technical Criteria

The Jailbreak client computing architecture not only recognizes traditional and existing clients, such as ATM networks, but also accommodates new clients. In fact, the Jailbreak client computing architecture is designed to scale and accommodate the Jailbreak application module based on the additional load that will be generated from growth in the number of clients. To accommodate this, all GEFS Jailbreak client platforms will be certified as "Runs on GEFS" clients to ensure that the client platform and associated GEFS applications are able to satisfy the GEFS Jailbreak client computing architecture. This certification of a specific client platform will ensure that the platform has exhibited the capacity characteristics required for each particular Jailbreak function. In particular, Jailbreak clients, especially DSS and OLAP clients will be able to deal with potentially large amounts of data.

b. Overview

The Jailbreak client computing architecture is based on a model in which existing, new, and emerging clients can seamlessly plug into the GEFS client computing architecture to effect retail banking financial transactions. The GEFS Jailbreak client computing architecture inherits and builds upon the secure, robust, and reliable environment provided by the middle and back-end tiers of the GEFS Jailbreak architecture. By leveraging the capabilities of the other tiers of the Jailbreak architecture, the GEFS Jailbreak client computing architecture quickly and cost-effectively provides for a wide variety of clients, which may securely transact retail banking functions. In addition, the Jailbreak client computing architecture promotes a zero administration client where possible. When management of the client is required, the Jailbreak client computing architecture uses remote management capabilities to mitigate the effect of local management requirements.

The Jailbreak client computing architecture supports a wide variety of clients, which can be classified as either Jailbreak existing and traditional clients or Jailbreak Java-based and Java-enabled clients. The Jailbreak client computing architecture supports each class of clients. This class-based support is referred to as the GEFS Jailbreak traditional client computing architecture and the GEFS Jailbreak Java client computing architecture. Examples of existing and traditional clients supported by the Jailbreak traditional client computing architecture include ATM networks, check services networks, and SWIFT. Examples of Java-based and Java-enabled client platforms supported by the Jailbreak Java Client Computing Architecture include Sun JavaStations, and PCs. The following describes the preferred implementation, based on the Jailbreak Java client computing architecture.

Figure 34 is a schematic representation of the clients available under the Jailbreak client computing architecture. The Jailbreak Java computing architecture is an integral part of the Jailbreak

client computing architecture. Indeed, the Jailbreak Java client computing architecture leverages specific technologies and products, such as Java, Sun JavaStations, BEA Jolt and BEA TUXEDO to provide a secure, robust and cost-effective platform for the retail banking environment. With the growth of clients, the Jailbreak client computing architecture leverages the BEA TUXEDO prioritization features to ensure that adequate response time is always provided. Most batch jobs are run when on-line transaction traffic is low, such as at night during non-prime banking hours. Accordingly, batch-oriented processes which run during prime banking hours will be given a lower priority than on-line transaction processing requests to ensure adequate response time. The Jailbreak client computing architecture supports the existing and traditional clients in a retail banking environment such as NACHA, SWIFT, ATM networks, POS networks, check service networks, credit card networks, debit card networks, image processing interfaces, and statement and document interfaces. Many of these existing systems and traditional client interfaces will be batch-based processes. In these cases, BEA TUXEDO prioritization features will be used to ensure adequate throughput and performance for these processes as well as concurrent on-line transactions.

As for the Java capabilities, in a preferred implementation, Jailbreak applications are developed to a 100% Pure Java standard. This will ensure a high degree of interoperability of Jailbreak applications on a wide range of Java-based and Java-enabled platforms. Each Jailbreak platform will be certified to ensure that its Java environment supports the 100% Pure Java Jailbreak applications. Accordingly, the GEFS Jailbreak Java client computing architecture is a multi-tier architecture in which Java-based applications are served from distributed middle-tier application servers. The GEFS Jailbreak Java client computing architecture is based on Java's "Write Once, Run Anywhere" computing model, which allows for the execution of the same Jailbreak applications on

multiple GEFS-certified client hardware platforms. The client-tier of the GEFS Jailbreak Java client computing architecture supports thin-client JavaStations, as well as traditional Windows 32-bit PCs.

Under the Jailbreak Java client computing architecture, Jailbreak applications reside on the middle-tier servers and can be downloaded on demand to any of the GEFS-supported clients. The presentation layer of the Jailbreak application presents the users with all of their needed retail banking functions with a Webtop graphical user interface. The client-tier of this architecture will be a lightweight, platform-neutral Java client, which inherits functionality from the secure, robust environment enabled by BEA TUXEDO systems. The Jailbreak Java applications also leverage capabilities of BEA Jolt to provide secure and robust transactional services for the retail-banking environment.

For Jailbreak Java applications, GEFS Jailbreak supports several client configurations within the Jailbreak Java client computing architecture. The Sun JavaStation with JavaOS and GEFS WebTop environment is the preferred client platform for the GEFS Jailbreak client. This platform is a GEFS-defined, controlled and managed fixed function device. The minimum configuration will likely be a JavaStation containing 32MB and a 14" Standard VGA monitor. The JavaStation is a fixed function and zero-administration client. However, to leverage a bank's recent PC client investment, the Java client computing architecture also supports specified Win32 PC configurations. The Jailbreak Java applications will be the same regardless of the client platform, but there will be additional administration, support, maintenance, and security issues relative to the GEFS-supported PC platform configurations.

JavaStations are the standard client platforms for Jailbreak clients. The JavaStation is configured as a fixed function retail banking device. The JavaStation uses standard PC-based components such as memory, monitors, and keyboards. JavaStations allow for manageability of the platform, should individual components need to be maintained.

JavaStations are expressly designed as a zero-administration client. JavaStations thus provide a major economic benefit compared to a traditional PC client platform. In particular, JavaStations will provide substantial economic benefits compared to traditional PCs, in the areas of administration and maintenance. For example, the JavaStation has no permanent local storage, and all information, including Jailbreak applications, is obtained over the network. The absence of local storage provides for improved security because data and applications always reside on the servers.

JavaStations run on the JavaOS, which is a small and efficient operating environment for the execution of Java applications directly on the JavaStations. JavaOS consists of a small Java kernel, an embedded Java Virtual Machine, Java windowing and graphics primitives, and networking protocols and device drivers written in Java. JavaOS is not a traditional operating system, and was designed as a compact, fast, easy-to-administer operating environment for pure Java applications.

For Jailbreak Java applications, a Webtop design is used for the Jailbreak GUI, one similar to Sun's HotJava views environment. The Jailbreak Webtop provides for an intuitive and easy to use graphical interface, specifically human-engineered for transaction workers in the retail banking environment. In a preferred implementation, a GEFS-branded WebTop, based on Sun's HotJava Views Product or Netscape Navigator will be used as the Web based container in which all jailbreak applications will execute. With this Webtop, the Pentium class Win32 PC and GEFS WebTop environment will be used on a branch-by-branch basis based upon specific bank branch requirements

for supporting existing non-GEFS applications. In one implementation, a minimum configuration for a GEFS supported client is anticipated to be: MS Windows 95 or NT 4.0 operating system, Pentium 90 or higher, 32 MB RAM or higher, and Ethernet 10/100MB/s NIC. With the Webtop, users can select among the various retail banking applications functions that they may perform by using an application selector to change the view of their WebTop to the appropriate Jailbreak function. The application selector's role as a simple point and click mechanism to switch between different Jailbreak functions facilitates ease of use and minimal training requirements.

c. Java Considerations

Figure 35 is a schematic representation of the software stick for a JavaStation. In contrast to the Java-enabled PC client platforms, the JavaStation does not require a separate installation of a Java virtual machine because the JavaOS contains a Java Virtual machine.

For JavaStations, in a preferred implementation, a GEFS-branded WebTop, based on Sun's HotJava Views Product or Navio's version of a 100% Java Netscape Navigator, will be used as the Web-based container in which all Jailbreak applications will execute. A Java-based MS Internet Explorer may also be used, if 100% Java. Essentially, JavaStations are similar to a SPARCStation.

As shown in Figure 35, JavaStations offer ease of configuration and remote manageability. JavaStations can be easily added to an existing branch's IP-based local network. Further, in a preferred implementation, Jailbreak JavaStations have field-replaceable devices that can be installed and enabled by simply powering the JavaStation after registering its network address with the GEFS data center. Thus, In the case of a JavaStation failure, a new unit can be shipped to a branch office and plugged into the GEFS network, without requiring any local branch IT support.

Figure 36 is a schematic representation of a smart card application module integrating with the architecture of the GEFS system. Despite the advantages of JavaStations, the Jailbreak PCs continue to use their recently acquired PCs for Jailbreak applications while continuing to use their PCs for non-Jailbreak local application processing. Thus, Jailbreak PCs also have their own advantages. For the Jailbreak PC, MS Windows 95 or MS Windows NT 4.0 is the base operating system. However, the Sun Java Virtual Machine for Win95 or WinNT based on JavaSoft jDK 1.1.1 or later, will be used to Java-enable the PC client platform. Further, a GEFS-branded WebTop, based on Sun's HotJava Views Product or Netscape Navigator (or possibly MS Internet Explorer), will be used as the Web-based container in which all Jailbreak applications will execute. Notably, Jailbreak-enabling an existing PC platform requires an additional administration and management burden, which is not incurred with the other Jailbreak supported platforms. GEFS will attempt to mitigate this burden by using automatic distribution mechanisms to revise and update all Jailbreak PC supported platforms to equivalent software releases.

Whether using Webstations or a Jailbreak PC, Jailbreak servers provide a combination of boot, network, print, Web and application services. Primarily, Jailbreak servers store data and serve the Jailbreak application. For network use and performance reasons, Jailbreak servers are physically located at branch locations. Each branch server is on the same subnet as the Jailbreak clients that it supports. To support the Jailbreak servers, a GEFS data center server provides back-up services over the WAN in case the branch server becomes unavailable. Although client performance would be degraded in the event that the branch server is unavailable, the ability of the branch to continue to operate using data center server services ensures high availability of Jailbreak application services at all times, even in the event of a failure.

In addition to storing and serving the Jailbreak applications, the Jailbreak servers also execute server-side components of the client side jailbreak applications. Some of the server side processes provided by the branch server include BEA Jolt Internet Relay services and other Web services. BEA Jolt Internet Relay services allow the server to relay BEA TUXEDO service requests on behalf of the Jailbreak clients. The use of this feature allows the BEA Jolt server components to be located in the data centers, and does not require the branch server to be a BEA TUXEDO node.

For the Jailbreak Java GUI, an easy and intuitive presentation layer is used for the business/logic layer of the Jailbreak system. The Jailbreak GUI supports both Java-based clients and Java-enabled clients. In particular, the Jailbreak GUI is human-engineered for the transaction-oriented retail banking user. As discussed above, in a preferred implementation, the Jailbreak GUI is WebTop.

WebTop provides the user intuitive push-button access to Jailbreak retail banking applications. WebTop is based on the concept of switched screens and viewing of single applications. The WebTop model of viewing only one application at a time is familiar to most retail banking users who currently use terminal applications with one function per screen and switch between screens to access different functions. The retail banking user will select from various Jailbreak application functions by using a GEFS Jailbreak selector. The selector is a GUI component that enables easy point-and-click access to specific jailbreak application functions.

As a unique feature of WebTop, this GUI provides a fixed set of retail branching applications based on the profile of a user who logs in. Thus, instead of relying on a retail banker's training and knowledge of arcane transaction names, WebTop uses the selector in Sun's HotJava Views product, an intuitive push-button GUI component, to switch between different applications and application

functions. WebTop provides a column of graphic icons on the left side of the screen representing the various Jailbreak retail banking functions the user may access. The selector allows the user to point and click on a graphical icon in the WebTop to access a particular Jailbreak function.

Additionally, WebTop provides other functionality within the Jailbreak applications, such as enabling or disabling the user's security and authority profiles. JavaStation clients require Network Information System Plus (NIS+) in order to participate as a Jailbreak client. Additionally, current versions of the JavaStation require a boot server to provide boot services. Jailbreak branch clients require the services of a Domain Name Services (DNS) server, and DNS allows jailbreak clients to access the names of systems and the route for connections outside of the directly-connected Ethernet network on which they reside. Although DNS will be provided by a centrally located DNS server in the data center, the Jailbreak branch server will be configured as a DNS slave to improve response time and network utilization.

In addition to the Java-specific components, the Jailbreak Java client computing architecture also includes BEA Jolt, which provides the Java applications with access to BEA TUXEDO services, such as application messaging, component management, and distributed transaction processing. BEA Jolt provides a Java-specific implementation of the BEA TUXEDO client library; thus, BEA Jolt acts as a proxy for a native BEA TUXEDO client. A Jolt server accepts requests from the Jailbreak Jolt clients and maps the requests into BEA TUXEDO service requests. A BEA TUXEDO service request, which originates from a Jolt client, is executed in the same manner as any other BEA TUXEDO request. The results of the BEA TUXEDO service request are returned to the Jolt Server that packages the results into a message that is sent to the Jolt client.

Notably, BEA Jolt is a new product. Yet BEA Jolt, by leveraging the proven capabilities of the BEA TUXEDO services, provides robust, modular, and scalable Java applications that address the transactional nature of the retail banking environment in Jailbreak applications. The Jailbreak branch server is also configured with the Jolt Internet Relay component. The Jolt Internet Relay component routes Jolt messages from a Jailbreak client to a Jolt server in the data center. The use of the Jolt Internet Relay alleviates the need for the Jolt server to run on the same branch server that is serving applications to the Jailbreak client. For security, the Jailbreak applications use a robust application security model in addition to leveraging Java's "sand box" security model. In addition, the Jailbreak security model also provides a secure Jailbreak client operational environment by using the security mechanisms provided by Jolt and an application security model.

As a final comment on Java in the Jailbreak application module, it should be noted that Jailbreak Java applications are referred to as a generic replacement for both applets and applications. The GEFS Jailbreak Java client computing architecture supports one or both types of executions or even possibly a hybrid solution. A Java applet and a Java application describe two different modes of start-up executions. A Java applet is a Java executable program that runs entirely within a Web browser window, and a Java applet may also be implemented within an external frame. This flexibility allows an applet to run independently of the browser within its own frame, but initially the applet must be loaded via an embedded link within a browser window. In contrast, a Java application is a Java applet that runs within is own frame but is not directly invoked from a browser window.

Whether a Jailbreak Java application more closely resembles an applet or an application, in a preferred implementation, all Jailbreak Java applications are based on Java's "Write Once, Run Anywhere" model of development and deployment. Thus, because the JavaStation platform implements a pure Java execution environment, Java applications that use other languages (accessed via Java's native methods or Active X components) will not execute in a JavaStation environment. As a result, all Jailbreak Java client applications are implemented using standard Java libraries and other 100% Pure Java components from third party vendors. By designing and developing all Jailbreak applications to run on the JavaStation platform, they are guaranteed to run on any other Java-enabled platform.

In contrast to PCs, JavaStations do not have a virtual memory system available by design. Therefore, Jailbreak applications manage real memory reasonably, likewise providing reasonable application response times. Current JavaStations can run with a maximum of 64 megabytes of memory. The JavaOS and HotJava Views environment will require approximately four megabytes of memory to boot and execute. In a preferred implementation, the remaining memory on JavaStations in the Jailbreak environment must be shared by all Jailbreak applications. These Jailbreak applications are designed to take into account the current state of memory on the client machine and allow Jailbreak applications to execute gracefully in low memory situations, such as when sharing resources with other applications.

If JavaStation is not used for a Jailbreak Java application, the Jailbreak Java application deployment for the PC platform will require the determination of a minimum PC configuration. For most PCs, this configuration includes a minimum of 32MB of memory.

Whether using a JavaStation or a PC, the architecture of individual Jailbreak Java applications still takes into account the download sequence and time required to load the appropriate jailbreak application functions for the different types of retail bank users. Criteria such as core log-on application sets, network topology, network utilization, and platform resource utilization are considered and balanced to achieve both reasonable start-up times and subsequent performance on the client platform. Similarly, loading all the classes for an application at start-up/log-in may be appropriate for a set of Jailbreak applications that may be loaded once a work session, such as each morning or shift, but for other sets of Jailbreak applications, where sessions may be short-lived, loading the entire Jailbreak application may not be appropriate. The Jailbreak applications are thus, depending on the circumstances, designed to allow for small segments of capability to be downloaded, followed by the loading of additional classes on demand, or slightly ahead of a user's need for the additional capability.

I. Application Module: Smart Card

1. Feasibility Analysis

Although smart cards have been used for quite some time in certain European countries, where hundreds of millions of cards are in circulation, their use in the rest of the world has been comparatively non-existent. There are various reasons for the lack of widespread global adoption of smart cards, including insufficient infrastructure, proprietary systems, and security issues. In the United States, this situation is changing rapidly, where market research shows that at least twenty-five percent of all Americans will be using smart cards by the year 2001. One technology market research firm predicts that smart cards will account for nearly half of the electronic commerce market by the year 2000, when the market is expected to total 7.3 billion dollars.

Although there are several smart card initiatives and pilot programs in effect in the United States, none of the existing smart card systems are likely to achieve a dominant position in the near future. The major reason for this is the proprietary approach of the existing smart card systems. Today, each smart card system involves closed and proprietary systems which are extremely inflexible. Each smart card system further contains a different software operating system, and the different smart card systems cannot operate easily with each other. More importantly, it is extremely difficult for these proprietary systems to interface with new and emerging electronic commerce channels. This unfortunate situation is unlikely to change in the near term due to the enormous investment that some smart card companies have made in developing their proprietary systems. Additionally, many of these companies view their closed systems as a method with which to control a segment of the smart card market and charge more for their services.

For these reasons, the smart card marketplace is primed for an open and extensible solution to providing and facilitating electronic commerce solutions with smart cards. In fact, smart card technology for a secure, flexible, robust and cost-effective client tier has recently emerged.

A GEFS electronic commerce application module, based on transactions initiated from GEFS smart cards, is another application module built on the overall GEFS concept. As implemented by the electronic commerce application module, the GEFS smart card module will have a natural and captive audience by marketing the service to existing retail-banking customers who are a completely untapped source of potential users of smart card technology. After all, a smart card module can be "plugged into" GEFS infrastructure 110, which is a natural fit, because a GEFS smart card system can build upon the inherent capabilities provided by the GEFS global electronic financial systems infrastructure. Building upon the foundation of the GEFS infrastructure allows the GEFS smart card

module to resolve many issues which have hampered early smart card systems. Further, although smart cards have been available, particularly in Europe, for the past few years, the adoption rate even in Europe has been slowed by the limitations of these systems. The GEFS smart card module overcomes the major limitations of flexibility, security, and reliability.

2. Overview

Figure 37 is a schematic representation of a smart card application module integrating with the architecture of the GEFS system. Figure 37 shows GEFS bank processing module 3710, which is connected to smart card database 3720. Customer 3730 interacts with GEFS bank processing module 3710 and may make transactions via smart card infrastructure provider 3740. With the present smart card systems, smart card applications have been developed in low-level machine code, known as assembler, which is specific to each smart card company and chip architecture. With these present systems, the smart card application running on the chip has been loaded on the card as part of the manufacturing process. As a result, present smart card applications have not been portable, and development processes have been both cumbersome and lengthy. Additionally, present smart cards have been 'hard-coded' in the sense that applications implemented by the smart card cannot be added or changed once the cards have been issued.

Unlike present smart card systems, the GEFS smart card application module is based on Java. With this added functionality, the GEFS smart card application module allows the smart card to be locked using a personal code so that only the card's owner can utilize the card. With the distinct flexibility and unique security of Java, the GEFS smart card application module is able to function equally well in the physical world of a shopping mall as in the virtual world of the Internet.

The GEFS smart card application module provides for a unique global payment system that combines the features of traditional cash currency with the convenience and security of electronic payment. The actual smart card is a tiny computer chip, running GEFS software, which is embedded in a GEFS branded plastic card. The GEFS smart card is based on an ISQ 7816 compliant integrated circuit card—the international standard for IC cards. By placing the physical computer chip on the GEFS smart card based on an established international standard, the smart card will be able to interface with a wide range of devices on a global basis. In addition, the integrated circuit on the smart card is designed to withstand the extremes of cold and heat, dampness, X-rays, or electrical interference.

In a preferred implementation, the GEFS smart card application module enables the smart card to store the electronic equivalent of cash on the card. The smart card is the size and weight of a traditional credit card. The integrated circuit on the card is programmed with GEFS smart card software to serve as an electronic purse. The electronic purse can be loaded with value, where it is stored until it is used as payment for goods or services at retailers or service outlets by inserting the card into a card reader. Thus, the smart card will allow for a wide range of electronic commerce activities. The user of a smart card can key trusted data on the card such as private digital key, transactional information, and of course electronic money.

With the GEFS smart card application module, there are essentially two types of applications made available. First, there are the applications that serve both the customer and the smart card directly. The customer will either put money onto the smart card or deposit the existing money from the smart card into a bank account. Second, is the application which involves the payment of fees to the smart card provider. In a preferred implementation, most of the necessary network

architecture will be contained within GEFS infrastructure 110. Indeed, requests for deposits or credits would generally come from one of three sources: the bank, the Internet, or the telephone. As these technologies are already integrated into the GEFS infrastructure 110, little additional network architecture is needed for the GEFS smart card applications. Indeed, all other smart card requirements would be readily served by the GEFS framework. For example, in the data center, a new database would simply be created to store the records concerning smart card transactions and ownership. However, the new database architecture would be minimal. For example, a typical smart card record would be linked to the customer's information already contained within GEFS system 200. Thus, the data associated with the smart card would be relatively small, because the majority of the information would be already stored within GEFS system 200. This lack of overhead further demonstrates the advantage of the GEFS smart card and application module.

V. CONCLUSION

It will thus be seen that the objects set forth above, among those made apparent from the preceding description, are efficiently attained, and because certain changes may be made in carrying out the above method and in the construction set forth, without departing from the spirit and scope of the invention, it is intended that all matter contained in the above description and shown in the accompanying drawings shall be interpreted as illustrative and not in a limiting sense.

WHAT IS CLAIMED IS:

1. A system for organizing a value network of a plurality of data centers, comprising:

an establishing component configured to establish a network center;

a connecting component configured to connect the network center to the data centers by a high-speed network;

a storing component configured to store data redundantly in each data center; and

a communication providing component configured to provide communication with the data centers via a wide area network.

2. The system of claim 1, wherein the communication providing component includes an access providing component configured to provide access to the network via a Web server.

3. The system of claim 2, wherein the access providing component further includes a using component configured to use a firewall on the Web server.

4. The system of claim 1, wherein the communication providing component includes an access providing component to provide access to the network via a bank client network.

5. The system of claim 1, wherein the communication providing component includes an access providing component to provide access to the network via a call center that allows telephone access to the network.

6. The system of claim 1, wherein the communication providing component includes an access providing component to provide access to the network via a kiosk that allows terminal access to the network.

7. The system of claim 1, wherein the communication providing component includes an access providing component to provide access to the network via a smart card.

8. A method of organizing a value network of a plurality of data centers, comprising the steps of:

establishing a network center;

connecting the network center to the data centers by a high-speed network;

redundantly storing data in each data center; and

providing communication with the data centers via a wide area network.

9. The method of claim 8, wherein the step of providing communication with the data centers via a wide area network includes the substep of

providing access to the network via a Web server.

10. The method of claim 9, wherein the step of providing access via the Web server further includes the substep of

using a firewall on the Web server.

11. The method of claim 8, wherein the step of providing communication with the data center via a wide area network includes the substep of

providing access to the network via a bank client network.

12. The method of claim 8, wherein the step of providing communication with the data center via a wide area network includes the substep of

providing access to the network via a call center that allows telephone access to the network.

13. The method of claim 8, wherein the step of providing communications with the data center via a wide area network includes the substep of

providing access to the network via a kiosk that allows terminal access to the network.

14. The method of claim 8, wherein the step of providing communications with the data center via a wide area network includes the substep of

providing access to the network via a smart card.

15. A system for organizing a value network of a plurality of data centers, comprising:

means for establishing a network center;

means for connecting the network center to the data centers by a high-speed network;

means for redundantly storing data in each data center; and

means for providing communication with the data centers via a wide area network.

16. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for organizing a value network of a plurality of data centers, the computer usable medium comprising:

an establishing module configured to establish a network center;

a connecting module configured to connect the network center to the data centers by a high-speed network;

a storing module configured to store data redundantly in each data center; and

a providing module configured to provide communication with the data centers via a wide area network.

17. A system for organizing a three-tier value network of a plurality of remote data centers, comprising:

an establishing component configured to establish an enterprise network management center;

a connecting component configured to connect the network management center to the remote data centers, each of the remote data centers having a presentation layer, an application layer, and an access layer;

a storing component configured to store data redundantly for each data center; and

a communication providing component configured to provide communication with the remote data centers via a wide area network.

18. The system of claim 17, wherein the communication providing component includes an access providing component configured to provide access to the network via a Web server.

19. The system of claim 18, wherein the access providing component further includes a using component configured to use a firewall on the Web server.

20. The system of claim 17, wherein the communication providing component includes an access providing component configured to provide access to the network via a bank client network.

21. The system of claim 17, wherein the communication providing component includes an access providing component configured to provide access to the network via a call center that allows telephone access to the network.

22. The system of claim 17, wherein the communication providing component includes an access providing component configured to provide access to the network via a kiosk that allows terminal access to the network.

23. The system of claim 22, wherein the access providing component further includes an interconnectivity providing component configured to provide interconnectivity to private access for electronic data interchange.

24. The system of claim 17, wherein the communication providing component includes an access providing component configured to provide access to the network via a smart card.

25. The system of claim 24, wherein the access providing component further includes an interconnectivity providing component configured to provide interconnectivity to private access for electronic data interchange.

26. The system of claim 17, wherein the communication providing component includes an interconnectivity providing component configured to provide interconnectivity to private access for electronic data interchange.

27. The system of claim 17, wherein the communication providing component includes an access providing component configured to provide access to the network via a remote service center.

28. The system of claim 17, wherein the wherein the communication providing component includes an access providing component configured to provide access to the network via a legacy service network.

29. A method of organizing a three-tier value network of a plurality of remote data centers, comprising the steps of:

establishing an enterprise network management center;

connecting the network management center to the remote data centers, each of the remote data centers having a presentation layer, an application layer, and an access layer;

redundantly storing data for each data center; and

providing communication with the remote data centers via a wide area network.

30. The method of claim 29, wherein the step of providing communication with the data centers via a wide area network includes the substep of

providing access to the network via a Web server.

31. The method of claim 30, wherein the step of providing access via the Web server further includes the substep of

using a firewall on the Web server.

32. The method of claim 29, wherein the step of providing communication with the data center via a wide area network includes the substep of

providing access to the network via a bank client network.

33. The method of claim 29, wherein the step of providing communication with the data center via a wide area network includes the substep of

providing access to the network via a call center that allows telephone access to the network.

34. The method of claim 29, wherein the step of providing communications with the data center via a wide area network further includes the substep of

providing access to the network via a kiosk that allows terminal access to the network.

35. The method of claim 34, wherein the step of providing access to the network via a kiosk further includes the substep of

providing interconnectivity to private access for electronic data interchange.

36. The method of claim 29, wherein the step of providing communications with the data center via a wide area network further includes the substep of

providing access to the network via a smart card.

37. The method of claim 36, wherein the step of providing access to the network via a smart card further includes the substep of

providing interconnectivity to private access for electronic data interchange.

38. The method of claim 29, wherein the step of providing communication with the remote data centers via a wide area network further includes the substep of

providing interconnectivity to private access for electronic data interchange.

39. The method of claim 29, wherein the step of providing communication with the remote data centers via a wide area network further includes the substep of

providing access via a remote service center.

40. The method of claim 29, wherein the step of providing communication with the remote data centers via a wide area network further includes the substep of

allowing access to the network via a legacy service network.

41. A system for organizing a three-tier value network of a plurality of remote data centers, comprising:

means for establishing an enterprise network management center;

means for connecting the network management center to the remote data centers, each of the remote data centers having a presentation layer, an application layer, and an access layer;

means for redundantly storing data for each data centers; and

means for providing communication with the remote data centers via a wide area network.

42. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for organizing a three-tier value network of a plurality of remote data centers, the computer usable medium comprising:

an establishing module configured to establish an enterprise network management center;

a connecting module configured to connect the network management center to the remote data centers, each of the remote data centers having a presentation layer, an application layer, and an access layer;

a storing module configured to store data redundantly for each data center; and

a providing module configured to provide communication with the remote data centers via a wide area network.

43. A system for organizing a presentation layer for a value network executed by a data processor, comprising:

a communicating component configured to communicate with a client on-line;

a batch file receiving component configured to receive a batch file from a client;

a request receiving component configured to receive a decision support request from the client;

an obtaining component configured to obtain data from a client; and

a presenting component configured to present a response to the client.

44. The system of claim 43, wherein the communicating component includes a providing component configured to provide interconnectivity to an Internet.

45. The system of claim 43, wherein the communicating component includes a providing component configured to provide interconnectivity to an Extranet.

46. The system of claim 43, wherein the communicating component includes a providing component configured to provide interconnectivity to an Intranet.

47. The system of claim 43, wherein the communicating component includes a providing component configured to provide interconnectivity to a bank teller workstation.

48. The system of claim 43, wherein the communicating component includes a providing component configured to provide interconnectivity to a voice response unit.

49. The system of claim 43, wherein the communicating component includes a providing component configured to provide interconnectivity to a smart card.

50. The system of claim 43, wherein the batch file receiving component includes a using component configured to use an ATM file.

51. The system of claim 43, wherein the communicating component includes a using component configured to use a JAVA-enabled program.

52. The system of claim 43, wherein the batch file receiving component includes a using component configured to use a UNIX-based process.

53. The system of claim 43, wherein the batch file receiving component includes a using component configured to use a Windows-based program.

54. A method of organizing a presentation layer for a value network executed by a data processor, comprising the steps of:

communicating with a client on-line;

receiving a batch file from a client;

receiving a decision support request from the client;

obtaining data from a client; and

presenting a response to the client.

55. The method of claim 54, wherein the step of communicating with a client includes the substep of

providing interconnectivity to an Internet.

56. The method of claim 54, wherein the step of communicating with a client includes the substep of

providing interconnectivity to an Extranet.

57. The method of claim 54, wherein the step of communicating with a client includes the substep of

providing interconnectivity to an Intranet.

58. The method of claim 54, wherein the step of communicating with a client includes the substep of

providing interconnectivity to a bank teller workstation.

59. The method of claim 54, wherein the step of communicating with a client includes the substep of

providing interconnectivity to a voice response unit.

60. The method of claim 54, wherein the step of communicating with a client includes the substep of

providing interconnectivity to a smart card.

61. The method of claim 54, wherein the step of receiving a batch file from a client includes the substep of

using an ATM file.

62. The method of claim 54, wherein the step of communicating with a client includes the substep of

using a JAVA-enabled program.

63. The method of claim 54, wherein the step of receiving a batch file from a client includes the substep of

using a UNIX-based process.

64. The method of claim 54, wherein the step of receiving a batch file from a client includes the substep of

using a Windows-based program.

65. A system for organizing a presentation layer for a value network executed by a data processor, comprising:

means for communicating with a client on-line;

means for receiving a batch file from a client;

means for receiving a decision support request from the client;

means for obtaining data from a client; and

means for presenting a response to the client.

66. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for organizing a presentation layer for a value network executed by a data processor, the computer usable medium comprising:

a communicating module configured to communicate with a client on-line;

a batch file receiving module configured to receive a batch file from a client;

a request receiving module configured to receive a decision support request from the client;

an obtaining module configured to obtain data from a client; and

a presenting module configured to present a response to the client.

67. A system for organizing an application logic layer for a value network, executed by a data processor, comprising:

a client communicating component configured to communicate with a client in the network using middleware;

an authenticating component configured to authenticate the client with a security database;

a processing component configured to process a request by the client pursuant to an application process; and

a replicating component configured to replicate the request over multiple communication channels.

68. The system of claim 67, further comprising an access allowing component configured to allow access to the network over an Internet.

69. The system of claim 68, wherein the access allowing component further includes a containing component configured to contain a JAVA applet for downloading to users.

70. The system of claim 67, wherein the replicating component includes a using component configured to use an on-line replication service.

71. The system of claim 67, wherein the replicating component includes a using component configured to use a queued replication service.

72. The system of claim 67, further comprising a supporting component configured to support a DSS process.

73. The system of claim 67, further comprising an external gateway providing component configured to provide an external gateway to the network.

74. The system of claim 73, wherein the external gateway providing component includes an access allowing component configured to allow access to the network via an incoming gateway.

75. The system of claim 73, wherein the external gateway providing component includes an access allowing component configured to allow access to the network via an outgoing gateway.

76. The system of claim 75, wherein the access allowing component further includes an online outgoing gateway communicating component configured to communicate over an online outgoing gateway.

77. The system of claim 75, wherein the access allowing component further includes a providing component configured to provide over a batch outgoing gateway.

78. A method of organizing an application logic layer for a value network, executed by a data processor, comprising the steps of:

communicating with a client in the network using middleware;

authenticating the client with a security database;

processing a request by the client pursuant to an application process; and

replicating the request over multiple communication channels.

79. The method of claim 78, further including the step of

allowing access to the network over an Internet.

80. The method of claim 79, wherein the step of allowing access to the network over the Internet further includes the substep of

containing a JAVA applet for downloading to users.

81. The method of claim 78, wherein the step of replicating the request over multiple communication channels of the network to ensure data durability further includes the substep of

using an on-line replication service.

82. The method of claim 78, wherein the step of replicating the request includes the substep of

using a queued replication service.

83. The method of claim 78, further including the step of

supporting a DSS process.

84. The method of claim 78, further including the step of

providing an external gateway to the network.

85. The method of claim 84, wherein the step of providing an external gateway to the network further includes the substep of

allowing access to the network via an incoming gateway.

86. The method of claim 84, wherein the step of providing an external gateway to the network further includes the substep of

allowing access to the network via an outgoing gateway.

87. The method of claim 86, wherein the step of allowing access to the network via an outgoing gateway further includes the substep of

communicating over an online outgoing gateway.

88. The method of claim 86, wherein the step of allowing access to the network via an outgoing gateway further includes the substep of

providing over a batch outgoing gateway.

89. A system for organizing an application logic layer for a value network, executed by a data processor, comprising:

means for communicating with a client in the network using middleware;

means for authenticating the client with a security database;

means for processing a request by the client pursuant to an application process; and

means for replicating the request over multiple communication channels.

90. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for organizing an application logic layer for a value network, executed by a data processor, the computer usable medium comprising:

a communicating module configured to communicate with a client in the network using middleware;

an authenticating module configured to authenticate the client with a security database;

a processing module configured to process a request by the client pursuant to an application process; and

a replicating module configured to replicate the request over multiple communication channels.

91. A system for organizing a data access layer for a value network, executed by a data processor, comprising:

a data storing component configured to store data for utilization by the network;

an interfacing component configured to interface with the data access layer to retrieve stored data;

a queuing component configured to queue a process for implementation by the network; and

a maintaining component configured to maintain a historical archive of images for processing by the network.

92. The system of claim 91 further comprising an image storing component configured to store images for utilization by the network; and

wherein the interfacing component is configured to interface with the data access layer to retrieve stored images.

93. The system of claim 91 wherein the data storing component includes a using component configured to use a redundant database management system (RDBMS).

94. The system of claim 92, wherein the image storing component further includes a using component configured to use an optical storage device.

95. A method of organizing a data access layer for a value network, executed by a data processor, comprising the steps of:

storing data for utilization by the network;

interfacing with the data access layer to retrieve stored data;

queuing a process for implementation by the network; and

maintaining a historical archive of images for processing by the network.

96. The method of claim 95 further including the step of

storing images for utilization by the network; and

wherein the step of interfacing includes the step of

interfacing with the data access layer to retrieve stored images.

97. The method of claim 95, wherein the step of storing data for utilization by the network further includes the substep of

using a redundant database management system (RDBMS).

98. The method of claim 96, wherein the step of storing images for utilization by the network further includes the substep of

using an optical storage device.

99. A system for organizing a data access layer for a value network, executed by a data processor, comprising:

means for storing data for utilization by the network;

means for interfacing with the data access layer to retrieve stored data;

means for queuing a process for implementation by the network; and

means for maintaining a historical archive of images for processing by the network.

100. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for organizing a data access layer for a value network, executed by a data processor, the computer usable medium comprising:

a storing module configured to store data for utilization by the network;

an interfacing module configured to interface with the data access layer to retrieve stored data;

a queuing module configured to queue a process for implementation by the network; and

a maintaining module configured to maintain a historical archive of images for processing by the network.

101. A system for processing a message by a client over a value network, executed by a data processor, comprising:

an initiating component configured to initiate a request by the client to join the network;

a submitting component configured to submit a message for an application;

a routing component configured to route the message to an application server;

an invoking component configured to invoke the message by the application server; and

a verifying component configured to verify a status regarding the condition of the message at the application server.

102. The system of claim 101, wherein the initiating component includes an appending component configured to append authentication/authorization data to the request.

103. The system of claim 102, wherein the appending component further includes an encryption component configured to encrypt the authentication/authorization data.

104. The system of claim 101, wherein the submitting component includes an encryption component configured to encrypt the message.

105. The system of claim 101, further comprising an examining component configured to examine the message for a routing requirement based on data content.

106. The system of claim 101, further comprising a queuing component configured to queue the message if the application server is busy.

107. The system of claim 101, further comprising a reinitializing component configured to reinitialize the network to receive a new request.

108. A method of processing a message by a client over a value network, executed by a data processor, comprising the steps of:

initiating a request by the client to join the network;

submitting a message for an application;

routing the message to an application server;

invoking the message by the application server; and

verifying a status regarding the condition of the message at the application server.

109. The method of claim 108, wherein the step of initiating a request by the client includes the substep of

appending authentication/authorization data to the request.

110. The method of claim 109, wherein the step of appending authentication/authorization data further includes the substep of

encrypting the authentication/authorization data.

111. The method of claim 108, wherein the step of submitting a message for an application further includes the substep of

encrypting the message.

112. The method of claim 108, further comprising the step of

examining the message for a routing requirement based on data content.

113. The method of claim 108, further including the step of

queuing the message if the application server is busy.

114. The method of claim 108, further including the step of

reinitializing the network to receive a new request.

115. A system for processing a message by a client over a value network, executed by a data processor, comprising:

means for initiating a request by the client to join the network;

means for submitting a message for an application;

means for routing the message to an application server;

means for invoking the message by the application server; and

means for verifying a status regarding the condition of the message at the application server.

116. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for processing a message by a client over a value network, executed by a data processor, the computer usable medium comprising:

an initiating module configured to initiate a request by the client to join the network;

a submitting module configured to submit a message for an application;

a routing module configured to route the message to an application server;

an invoking module configured to invoke the message by the application server; and

a verifying module configured to verify a status regarding the condition of the message at the application server.

117. A system for organizing a data center on a value network comprising:

an application server providing component configured to provide an application server for applications used by the value network;

a replica application server providing component configured to provide a replica application server for applications used by the value network;

a database server providing component configured to provide a database server for data used in the value network;

a replica database server providing component configured to provide a replica database server for data utilized by the value network;

a communications server providing component configured to provide a communications server for the interconnecting of servers; and

a connecting component configured to connect the servers over a network.

118. The system of claim 117, further comprising an image database server providing component configured to provide an image database server.

119. The system of claim 117, further comprising an archive database server providing component configured to provide an archive database server.

120. The system of claim 117, further comprising a management/monitoring station providing component configured to provide a management/monitoring station.

121. The system of claim 117, further comprising a gateway hardware providing component configured to provide gateway hardware.

122. The system of claim 117, further comprising a printing services providing component configured to provide printing services for the data center.

123. The system of claim 121, wherein the gateway hardware providing component includes a using component configured to use a router to access a legacy network.

124. The system of claim 121, wherein the gateway hardware providing component includes an Internet router using component configured to use an Internet router.

125. The system of claim 124, wherein the Internet router using component further includes a firewall using component configured to use a firewall.

126. The system of claim 125, wherein the Internet router using component further includes a frame relay switch using component configured to use a frame relay switch for allowing access to frame relay networks.

127. A method of organizing a data center on a value network, comprising the steps of:

providing an application server for applications used by the value network;

providing a replica application server for applications used by the value network;

providing a database server for data used in the value network;

providing a replica database server for data utilized by the value network;

providing a communications server for the interconnecting of servers; and

connecting the servers over a network.

128. The method of claim 127, further comprising the step of

providing an image database server.

129. The method of claim 127, further comprising the step of

providing an archive database server.

130. The method of claim 127, further comprising the step of

providing a management/monitoring station.

131. The method of claim 127, further comprising the step of

providing gateway hardware.

132. The method of claim 127, further comprising the step of

providing printing services for the data center.

133. The method of claim 131, wherein the step of providing gateway hardware further includes the substep of

using a router to access a legacy network.

134. The method of claim 131, wherein the step of providing gateway hardware further includes the substep of

using an Internet router.

135. The method of claim 134, wherein the step of using an Internet router further includes the substep of

using a firewall.

136. The method of claim 135, wherein the step of using a router further includes the substep of using a frame relay switch for allowing access to frame relay networks.

137. A system for organizing a data center on a value network comprising:

means for providing an application server for applications used by the value network;

means for providing a replica application server for applications used by the value network;

means for providing a database server for data used in the value network;

means for providing a replica database server for data utilized by the value network;

means for providing a communications server for the interconnecting of servers; and

means for connecting the servers over a network.

138. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for organizing a data center on a value network, the computer usable medium comprising:

a providing module configured to provide an application server for applications used by the value network;

a providing module configured to provide a replica application server for applications used by the value network;

a providing module configured to provide a database server for data used in the value network;

a providing module configured to provide a replica database server for data utilized by the value network;

a providing module configured to provide a communications server for the interconnecting of servers; and

a connecting module configured to connect the servers over a network.

139. A system for organizing a data center on a value network, comprising:

a first application server providing component configured to provide a first application server for applications used by the value network;

a second application server providing component configured to provide a second application server for applications used by the value network;

a first replica application server providing component configured to provide a first replica application server for applications used by the value network;

a second replica application server providing component configured to provide a second replica application server for applications used by the value network;

a first database server providing component configured to provide a first database server for data used by the value network;

a second database server providing component configured to provide a second database server for data used by the value network;

a first replica database server providing component configured to provide a first replica database server for data used by the value network;

a second replica database server providing component configured to provide a second replica database server for data used by the value network;

an image database server providing component configured to provide an image database server for image data used by the value network;

a replica image database providing component configured to provide a replica image database server for image data used by the value network;

a historical database server providing component configured to provide a historical database server for archiving data from the value network;

a replica historical database server providing component configured to provide a replica historical database server for archiving data from the value network;

a communications server providing component configured to provide a communications server for the interconnecting of servers;

a replica communications server providing component configured to provide a replica communications server for the interconnecting of servers; and

an Ethernet connecting component configured to connect the servers over a fast Ethernet connection.

140. The system of claim 139, wherein the communications server providing component includes a using component configured to use a router for connecting to other data centers.

141. The system of claim 140, wherein the using component further includes a utilizing component configured to utilize a frame relay switch for connecting to frame relay sources.

142. The system of claim 139, further comprising a parallel server pair connecting component configured to connect the first database server and the second database server as a parallel server pair.

143. The system of claim 142, wherein parallel server pair connecting component includes a gateway router connecting component configured to connect a gateway router to the first and second database parallel server pair.

144. A method of organizing a data center on a value network, comprising the steps of:

providing a first application server for applications used by the value network;

providing a second application server for applications used by the value network;

providing a first replica application server for applications used by the value network;

providing a second replica application server for applications used by the value network;

providing a first database server for data used by the value network;

providing a second database server for data used by the value network;

providing a first replica database server for data used by the value network;

providing a second replica database server for data used by the value network;

providing an image database server for image data used by the value network;

providing a replica image database server for image data used by the value network;

providing a historical database server for archiving data from the value network;

providing a replica historical database server for archiving data from the value network;

providing a communications server for the interconnecting of servers;

providing a replica communications server for the interconnecting of servers; and

connecting the servers over a fast Ethernet connection.

145. The method of claim 144, wherein the step of providing a communications server for the interconnection of servers further includes the substep of

using a router for connecting to other data centers.

146. The method of claim 145, wherein the substep of using a router for connecting to other data centers further includes the substep of

utilizing a frame relay switch for connecting to frame relay sources.

147. The method of claim 144, further including the step of

connecting the first database server and the second database server as a parallel server pair.

148. The method of claim 147, wherein the step of connecting the first database server and the second database server includes the substep of

connecting a gateway router to the first and second database parallel server pair.

149. A system for organizing a data center on a value network, comprising:

means for providing a first application server for applications used by the value network;

means for providing a second application server for applications used by the value network;

means for providing a first replica application server for applications used by the value network;

means for providing a second replica application server for applications used by the value network;

means for providing a first database server for data used by the value network;

means for providing a second database server for data used by the value network;

means for providing a first replica database server for data used by the value network;

means for providing a second replica database server for data used by the value network;

means for providing an image database server for image data used by the value network;

means for providing a replica image database server for image data used by the value network;

means for providing a historical database server for archiving data from the value network;

means for providing a replica historical database server for archiving data from the value network;

means for providing a communications server for the interconnecting of servers;

means for providing a replica communications server for the interconnecting of servers; and

means for connecting the servers over a fast Ethernet connection.

150. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for organizing a data center on a value network, the computer usable medium comprising:

a first application server providing module configured to provide a first application server for applications used by the value network;

a second application server providing module configured to provide a second application server for applications used by the value network;

a first replica application server providing module configured to provide a first replica application server for applications used by the value network;

a second replica application server providing module configured to provide a second replica application server for applications used by the value network;

a first database server providing module configured to provide a first database server for data used by the value network;

a second database server providing module configured to provide a second database server for data used by the value network;

a first replica database server providing module configured to provide a first replica database server for data used by the value network;

a second replica database server providing module configured to provide a second replica database server for data used by the value network;

an image database server providing module configured to provide an image database server for image data used by the value network;

a replica image database providing module configured to provide a replica image database server for image data used by the value network;

a historical database server providing module configured to provide a historical database server for archiving data from the value network;

a replica historical database server providing module configured to provide a replica historical database server for archiving data from the value network;

a communications server providing module configured to provide a communications server for the interconnecting of servers;

a replica communications server providing module configured to provide a replica communications server for the interconnecting of servers; and

a connecting module configured to connect the servers over a fast Ethernet connection.

151. A system for organizing a value network for core retail banking applications, comprising:

a branch server-router connecting component configured to connect a branch server to a router;

a network management-router connecting component configured to connect the router to a network management center;

an allowing component configured to allow connectivity from the branch server to the value network; and

a permitting component configured to permit access to the branch server by a network management center connected to the value network.

152. The system of claim 151, further comprising the step of branch server-gateway connecting component configured to connect the branch server to an Internet gateway via the network management center.

153. The system of claim 151, further comprising a branch server-legacy connecting component configured to connect the branch server to a legacy network via the network management center.

154. The system of claim 153, wherein the branch server-legacy connecting component includes an accessing component configured to access an automated clearing house (ACH).

155. The system of claim 151, further comprising a teller branch server-branch client connecting component configured to connect the branch server to a branch client for a teller.

156. The system of claim 151, further comprising a loan officer branch server-branch client connecting component configured to connect the branch server to a branch client for a loan officer.

157. The system of claim 151, further comprising a branch manager branch server-branch client connecting component configured to connect the branch server to a branch client for a branch manager.

158. A method of organizing a value network for core retail banking applications, comprising the steps of:

connecting a branch server to a router;

connecting the router to a network management center;

allowing connectivity from the branch server to the value network; and

permitting access to the branch server by a network management center connected to the value network.

159. The method of claim 158, further comprising the step of

connecting the branch server to an Internet gateway via the network management center.

160. The method of claim 158, further comprising the step of

connecting the branch server to a legacy network via the network management center.

161. The method of claim 160, wherein the step of connecting the branch server to a legacy network includes the substep of

accessing an automated clearing house (ACH).

162. The method of claim 158, further comprising the step of

connecting the branch server to a branch client for a teller.

163. The method of claim 158, further comprising the step of

connecting the branch server to a branch client for a loan officer.

164. The method of claim 158, further comprising the step of

connecting the branch server to a branch client for a branch manager;

165. A system for organizing a value network for core retail banking applications, comprising:

means for connecting a branch server to a router;

means for connecting the router to a network management center;

means for allowing connectivity from the branch server to the value network; and

means for permitting access to the branch server by a network management center connected to the value network.

166. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for organizing a value network for core retail banking applications, the computer usable medium comprising:

a connecting module configured to connect a branch server to a router;

a connecting module configured to connect the router to a network management center;

an allowing module configured to allow connectivity from the branch server to the value network; and

a permitting module configured to permit access to the branch server by a network management center connected to the value network.

167. A system for interfacing a smart card with a value network, comprising:

an embedding component configured to embed within the smart card a computer chip for retaining customer information;

an encoding component configured to encode the computer chip with customer information;

an allowing component configured to allow interaction with the smart card by the value network; and

a permitting component configured to permit access to the smart card by a network management center connected to the value network.

168. The system of claim 167, wherein the embedding component includes a using component configured to use an ISO 7816 compliant integrated circuit card.

169. The system of claim 167, wherein the allowing component includes a using component configured to use JavaSoft.

170. The system of claim 167, wherein the allowing component includes an interacting component configured to interact with the card at a bank.

171. The system of claim 167, wherein the allowing component includes an interacting component configured to interact with the card via an Internet.

172. The system of claim 167, wherein the allowing component includes an interacting component configured to interact with the card via a telephone.

173. The system of claim 167, wherein the encoding component includes a using component configured to use encryption.

174. The system of claim 167, wherein the allowing component includes an interacting component configured to interact with the card via an Extranet.

175. The system of claim 167, wherein the allowing component includes an interacting component configured to interact with the card via an Intranet.

176. A method of interfacing a smart card with a value network, comprising the steps of:

embedding within the smart card a computer chip for retaining customer information;

encoding the computer chip with customer information;

allowing interaction with the smart card by the value network; and

permitting access to the smart card by a network management center connected to the value network.

177. The method of claim 176, wherein the step of embedding a computer chip includes the substep of

using an ISO 7816 compliant integrated circuit card.

178. The method of claim 176, wherein the step of allowing interaction with the computer chip further includes the substep of

using JavaSoft.

179. The method of claim 176, wherein the step of allowing interaction with the smart card by the value network further includes the substep of

interacting with the card at a bank.

180. The method of claim 176, wherein the step of allowing interaction with the smart card by a value network further includes the substep of

interacting with the card via an Internet.

181. The method of claim 176, wherein the step of allowing interaction with the smart card by a value network further includes the substep of

interacting with the card via a telephone.

182. The method of claim 176, wherein the step of encoding the computer chip with customer information further includes the substep of

using encryption.

183. The method of claim 176, wherein the step of allowing interaction with the smart card by a value network further includes the substep of

interacting with the card via an Extranet.

184. The method of claim 176, wherein the step of allowing interaction with the smart card by a value network further includes the substep of

interacting with the card via an Intranet.

185. A system for interfacing a smart card with a value network, comprising:

means for embedding within the smart card a computer chip for retaining customer information;

means for encoding the computer chip with customer information;

means for allowing interaction with the smart card by the value network; and

means for permitting access to the smart card by a network management center connected to the value network.

186. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for interfacing a smart card with a value network, the computer usable medium comprising:

an embedding module configured to embed within the smart card a computer chip for retaining customer information;

an encoding module configured to encode the computer chip with customer information;

an allowing module configured to allow interaction with the smart card by the value network; and

a permitting module configured to permit access to the smart card by a network management center connected to the value network.

187. A system for processing a financial transaction over a network, comprising:

a presentation logic transmitting component configured to transmit presentation logic from an application server to a presentation client for displaying a transaction request interface, the application server having business rules associated with a financial transaction, and the presentation client having interface controls;

a receiving component configured to receive financial data for said transaction request interface through the interface controls, said transaction request interface including an indicia of the type of financial transaction requested and identifying information associating the financial transaction with the user account;

a transaction request transmitting component configured to transmit said transaction request interface over the network to the application server;

a validating component configured to validate said financial data in the application server according to the associated business rules;

a processing component configured to process the financial transaction corresponding to the indicia in accordance with the associated business rules; and

an accessing component configured to access the data center selectively for reading and writing the financial data to the relational databases and processing the financial transaction.

188. The system of claim 187, further comprising a parsing component configured to parse said financial transaction indicia and said identifying information from said transaction request interface.

189. The system of claim 187, further comprising a presentation logic updating component configured to update said presentation logic dynamically over the network.

190. The system of claim 187, further comprising a non-interrupting updating component configured to update the business rules and presentation logic dynamically without interrupting the network.

191. The system of claim 187, further comprising a providing component configured to provide a three-tier, client server electronic network system, said three tiers including a data center having

relational databases comprising at least user account data, said three tiers being coupled by an electronic network.

192. The system of claim 187, further comprising a selecting component configured to select financial data in accordance with the financial transaction.

193. The system of claim 187, wherein the processing component includes a transmitting component configured to transmit an error notification from the application server to the presentation client if said identifying information is not validated.

194. The system of claim 187, wherein the processing component further comprises

a generating component configured to generate a results indication identifying a completion status of the financial transaction;

a results transmitting component configured to transmit the results indication from the application server to the presentation client; and

a displaying component configured to display the results indication on the presentation client in accordance with the presentation logic.

195. A method of processing a financial transaction over a network, comprising the steps of:

transmitting presentation logic from an application server to a presentation client for displaying a transaction request interface, the application server having business rules associated with a financial transaction, and the presentation client having interface controls;

receiving financial data for said transaction request interface through the interface controls, said transaction request interface including an indicia of the type of financial transaction requested and identifying information associating the financial transaction with the user account;

transmitting said transaction request interface over the network to the application server;

validating said financial data in the application server according to the associated business rules;

processing the financial transaction corresponding to the indicia in accordance with the associated business rules; and

selectively accessing the data center for reading and writing the financial data to the relational databases and processing the financial transaction.

196. The method of claim 195, further comprising the step of

parsing said financial transaction indicia and said identifying information from said transaction request interface.

197. The method of claim 195, further comprising the step of

dynamically updating said presentation logic over the network.

198. The method of claim 195, further comprising the step of

dynamically updating the business rules and presentation logic without interrupting the network.

199. The method of claim 195, further comprising the step of

providing a three-tier, client server electronic network system, said three tiers including a data center having relational databases comprising at least user account data, said three tiers being coupled by an electronic network.

200. The method of claim 195, further comprising the step of

selecting financial data in accordance with the financial transaction.

201. The method of claim 195, wherein the step of processing a financial transaction further comprises the substep of

transmitting an error notification from the application server to the presentation client if said identifying information is not validated.

202. The method of claim 195, wherein the step of processing a financial transaction further comprises the substeps of

generating a results indication identifying a completion status of the financial transaction;

transmitting the results indication from the application server to the presentation client; and

displaying the results indication on the presentation client in accordance with the presentation logic.

203. A system for processing a financial transaction over a network, comprising:

means for transmitting presentation logic from an application server to a presentation client for displaying a transaction request interface, the application server having business rules associated with a financial transaction, and the presentation client having interface controls;

means for receiving financial data for said transaction request interface through the interface controls, said transaction request interface including an indicia of the type of financial transaction requested and identifying information associating the financial transaction with the user account;

means for transmitting said transaction request interface over the network to the application server;

means for validating said financial data in the application server according to the associated business rules;

means for processing the financial transaction corresponding to the indicia in accordance with the associated business rules; and

means for selectively accessing the data center for reading and writing the financial data to the relational databases and processing the financial transaction.

204. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for processing a financial transaction over a network, the computer usable medium comprising:

a presentation logic transmitting module configured to transmit presentation logic from an application server to a presentation client for displaying a transaction request interface, the application server having business rules associated with a financial transaction, and the presentation client having interface controls;

a receiving module configured to receive financial data for said transaction request interface through the interface controls, said transaction request interface including an indicia of the type of financial transaction requested and identifying information associating the financial transaction with the user account;

a transaction request transmitting module configured to transmit said transaction request interface over the network to the application server;

a validating module configured to validate said financial data in the application server according to the associated business rules;

a processing module configured to process the financial transaction corresponding to the indicia in accordance with the associated business rules; and

an accessing module configured to access the data center selectively for reading and writing the financial data to the relational databases and processing the financial transaction.

205. A system for interfacing application modules with a value network, comprising:

an associating component configured to associate an application module with a data center connected to the value network;

an operation providing component configured to provide for operation of the application module by the data center; and

a permitting component configured to permit access to the application module by a network management center connected to the value network.

206. The system of claim 205, further comprising an interconnecting component configured to interconnect the application module with the data center.

207. The system of claim 205, further comprising an establishing component configured to establish a communications link between the network management center and the data center.

208. The system of claim 205, wherein the operation providing component includes a core retail banking functions providing component configured to provide a set of operations for performing core retail banking functions.

209. The system of claim 205, wherein the operation providing component includes a commercial banking functions providing component configured to provide a set of operations for performing commercial banking functions.

210. The system of claim 205, wherein the operation providing component includes a wholesale banking functions providing component configured to provide a set of operations for performing wholesale banking functions.

211. The system of claim 205, wherein the operation providing component includes a direct banking functions providing component configured to provide a set of operations for performing direct banking functions.

212. The system of claim 205, wherein the operation providing component includes an electronic compliance reporting providing component configured to provide a set of operations for performing electronic compliance reporting.

213. The system of claim 205, wherein the operation providing component includes a browser-based clearing providing component configured to provide a set of operations for performing browser-based clearing.

214. The system of claim 205, wherein the operation providing component includes a virtual private extranet banking providing component configured to provide a set of operations for performing virtual private extranet banking.

215. The system of claim 205, wherein the operation providing component includes an elite private banking providing component configured to provide a set of operations for performing elite private banking.

216. The system of claim 205, wherein the operation providing component includes a brokerage and securities providing component configured to provide a set of operations for performing brokerage and securities.

217. The system of claim 205, wherein the operation providing component includes an advanced analytics providing component configured to provide a set of operations for performing advanced analytics.

218. The system of claim 205, wherein the operation providing component includes a digital credit card processing providing component configured to provide a set of operations for performing digital credit card processing.

219. The system of claim 205, wherein the operation providing component includes a reinsurance applications providing component configured to provide a set of operations for performing reinsurance applications.

220. The system of claim 205, wherein the operation providing component includes a call center providing component configured to provide a set of operations for performing a call center.

221. The system of claim 205, wherein the operation providing component includes a Year 2000 solution providing component configured to provide a set of operations for performing a Year 2000 solution.

222. The system of claim 205, wherein the operation providing component includes a Global 1000 processing providing component configured to provide a set of operations for performing Global 1000 processing.

223. A method of interfacing application modules with a value network, comprising the steps of:

associating an application module with a data center connected to the value network;

providing for operation of the application module by the data center; and

permitting access to the application module by a network management center connected to the value network.

224. The method of claim 223, further comprising the step of

interconnecting the application module with the data center.

225. The method of claim 223, further comprising the step of

establishing a communications link between the network management center and the data center.

226. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing core retail banking functions.

227. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing commercial banking functions.

228. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing wholesale banking functions.

229. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing direct banking functions.

230. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing electronic compliance reporting.

231. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing browser-based clearing.

232. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing virtual private extranet banking.

233. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing elite private banking.

234. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing brokerage and securities.

235. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing advanced analytics.

236. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing digital credit card processing.

237. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing reinsurance applications.

238. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing a call center.

239. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing a Year 2000 solution.

240. The method of claim 223, wherein the step of providing for operation of the application module by the data center further includes the substep of

providing a set of operations for performing Global 1000 processing.

241. A system for interfacing application modules with a value network, comprising:

means for associating an application module with a data center connected to the value network;

means for providing for operation of the application module by the data center; and

means for permitting access to the application module by a network management center connected to the value network.

242. A computer program product comprising:

a computer usable medium having computer readable code embodied therein for interfacing application modules with a value network, the computer usable medium comprising:

an associating module configured to associate an application module with a data center connected to the value network;

an operation providing module configured to provide for operation of the application module by the data center; and

a permitting module configured to permit access to the application module by a network management center connected to the value network.

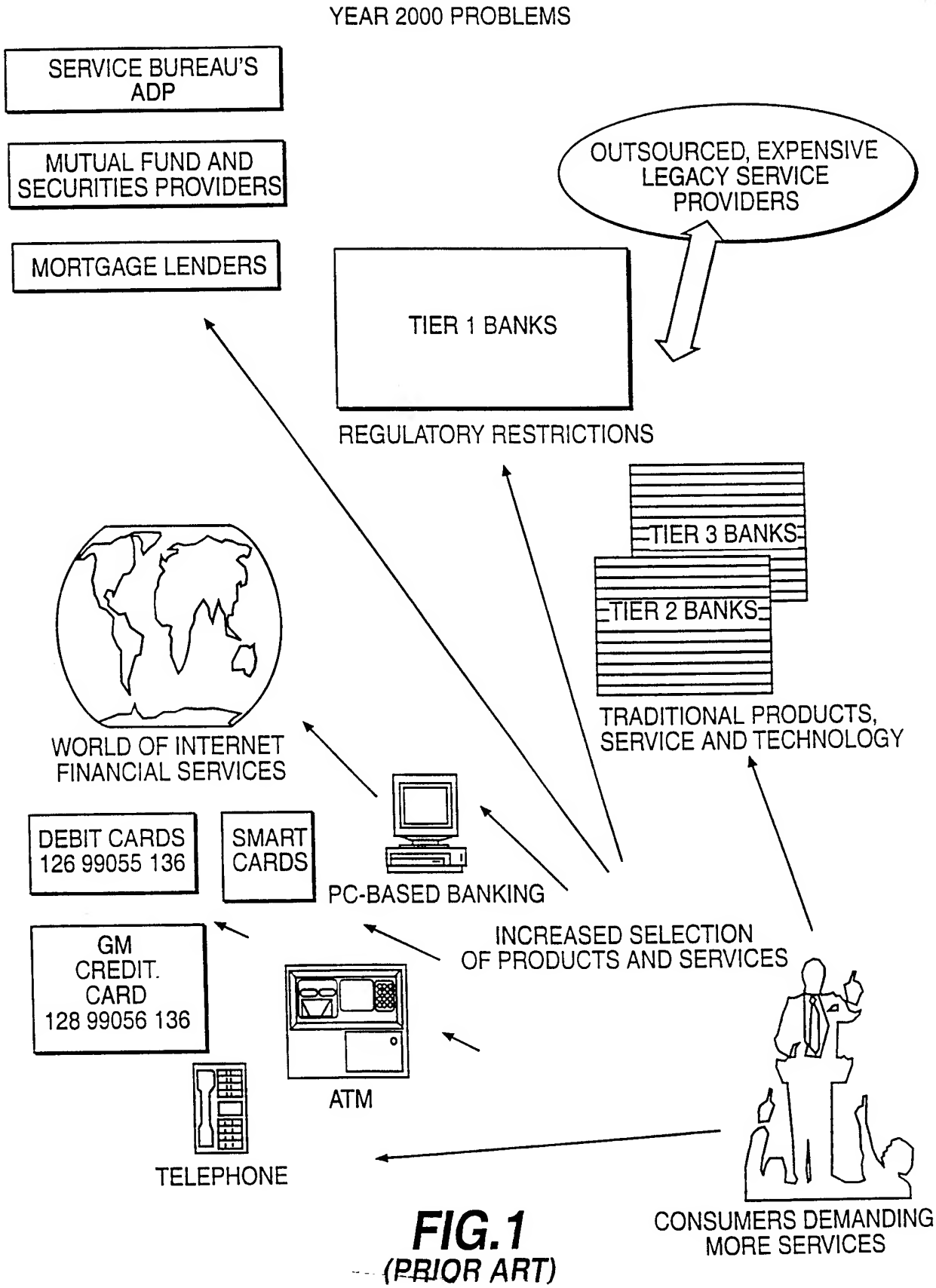


FIG.1
(PRIOR ART)

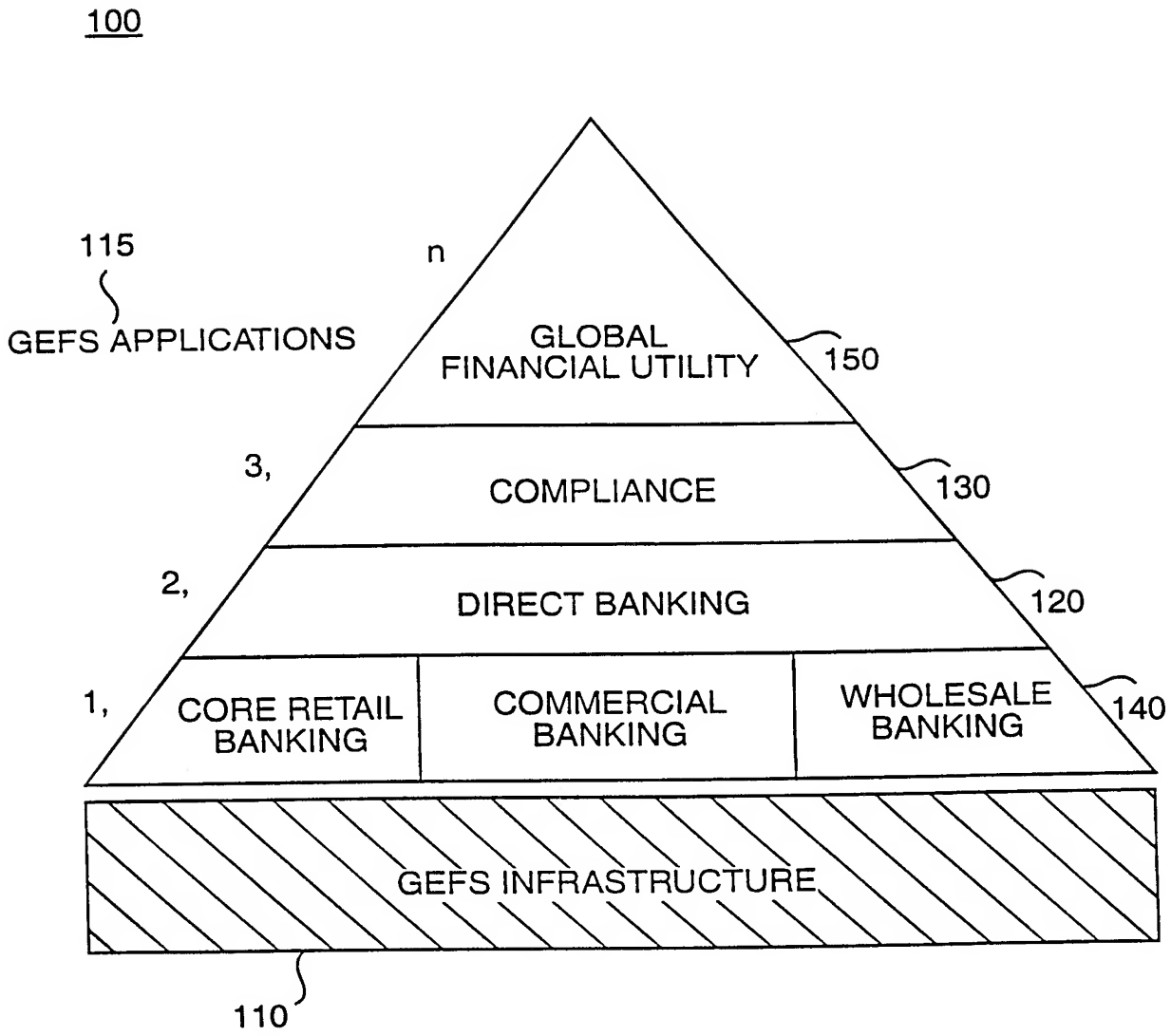

100
115
GEFS APPLICATIONS
n
GLOBAL FINANCIAL UTILITY
150
3,
COMPLIANCE
130
2,
DIRECT BANKING
120
1,
CORE RETAIL BANKING
COMMERCIAL BANKING
WHOLESALE BANKING
140
GEFS INFRASTRUCTURE
110

FIG. 2A

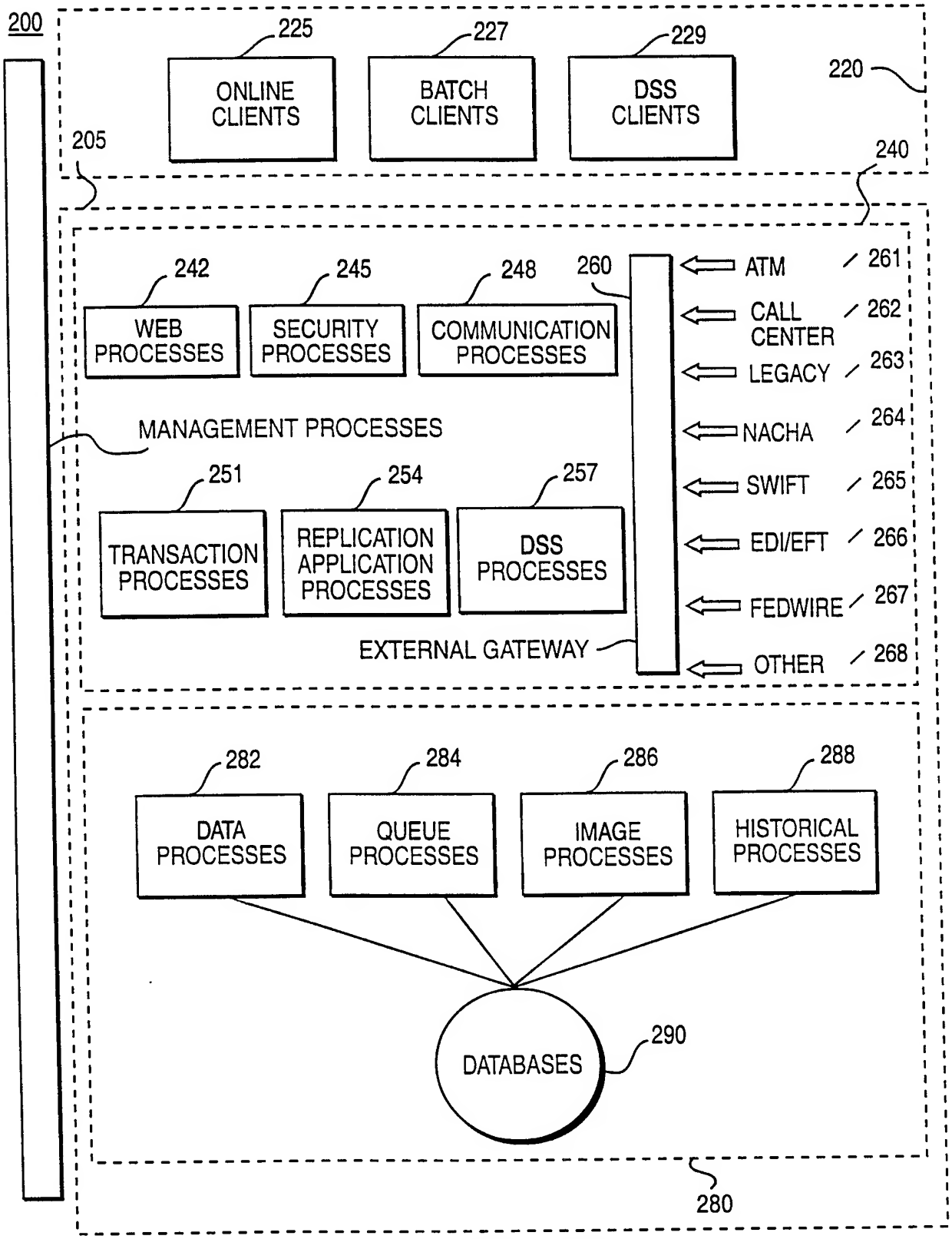

200
225
ONLINE CLIENTS
227
BATCH CLIENTS
229
DSS CLIENTS
220
205
240
242
WEB PROCESSES
245
SECURITY PROCESSES
248
COMMUNICATION PROCESSES
260
ATM
261
CALL CENTER
262
LEGACY
263
NACHA
264
SWIFT
265
EDI/EFT
266
FEDWIRE
267
OTHER
268
MANAGEMENT PROCESSES
251
TRANSACTION PROCESSES
254
REPLICATION APPLICATION PROCESSES
257
DSS PROCESSES
EXTERNAL GATEWAY
282
DATA PROCESSES
284
QUEUE PROCESSES
286
IMAGE PROCESSES
288
HISTORICAL PROCESSES
DATABASES
290
280

FIG. 2B

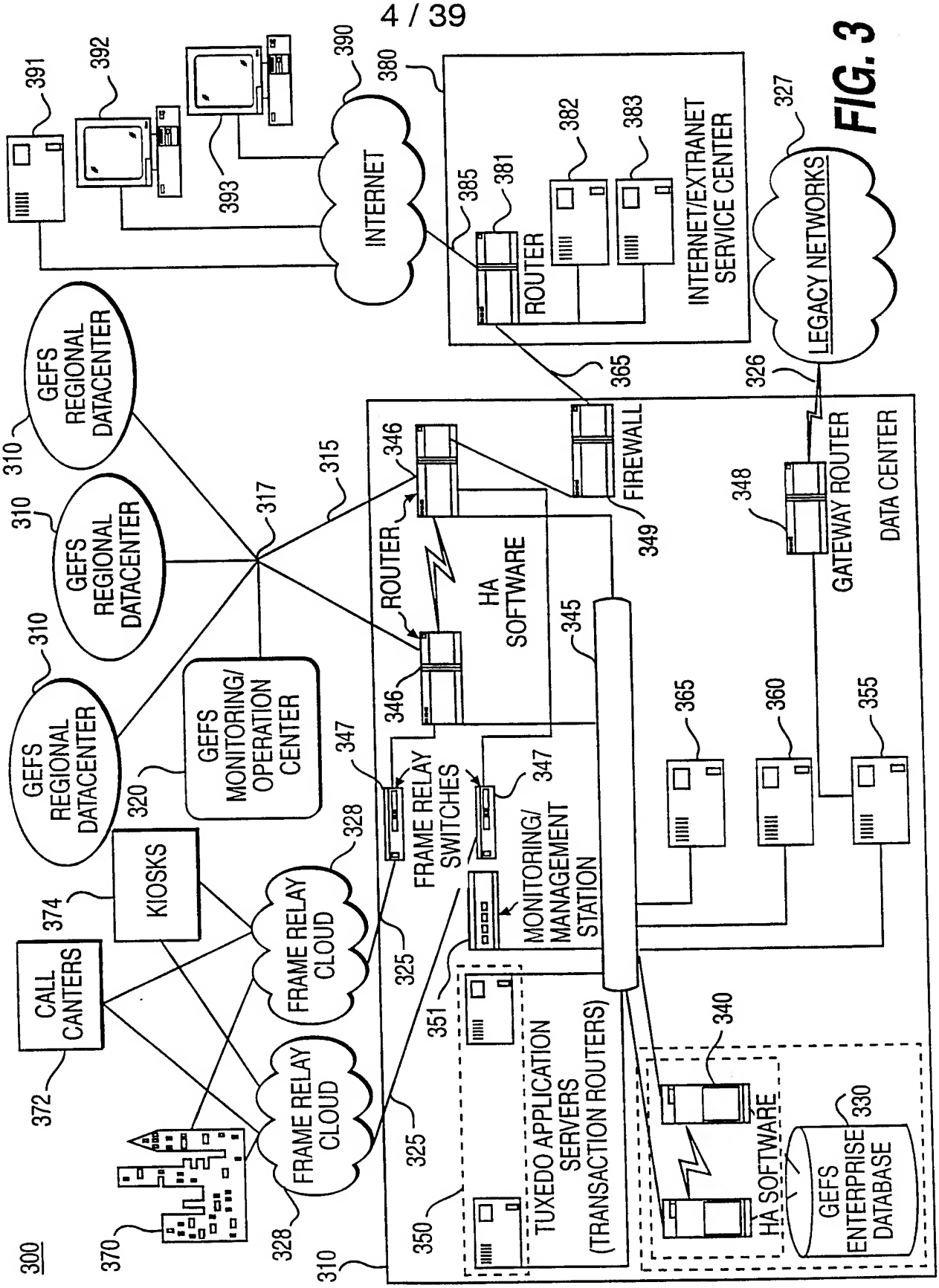

300
310
GEFS REGIONAL DATACENTER
GEFS REGIONAL DATACENTER
GEFS REGIONAL DATACENTER
GEFS MONITORING/ OPERATION CENTER
320
KIOSKS
374
CALL CANTERS
372
FRAME RELAY CLOUD
328
FRAME RELAY CLOUD
328
370
317
315
INTERNET
390
391
392
393
380
ROUTER
385
381
382
383
INTERNET/EXTRANET SERVICE CENTER
327
LEGACY NETWORKS
326
365
FIREWALL
349
346
ROUTER
346
HA SOFTWARE
345
348
GATEWAY ROUTER
DATA CENTER
FRAME RELAY SWITCHES
347
347
MONITORING/ MANAGEMENT STATION
325
325
351
TUXEDO APPLICATION SERVERS (TRANSACTION ROUTERS)
350
340
HA SOFTWARE
GEFS ENTERPRISE DATABASE
330
365
360
355

FIG. 3

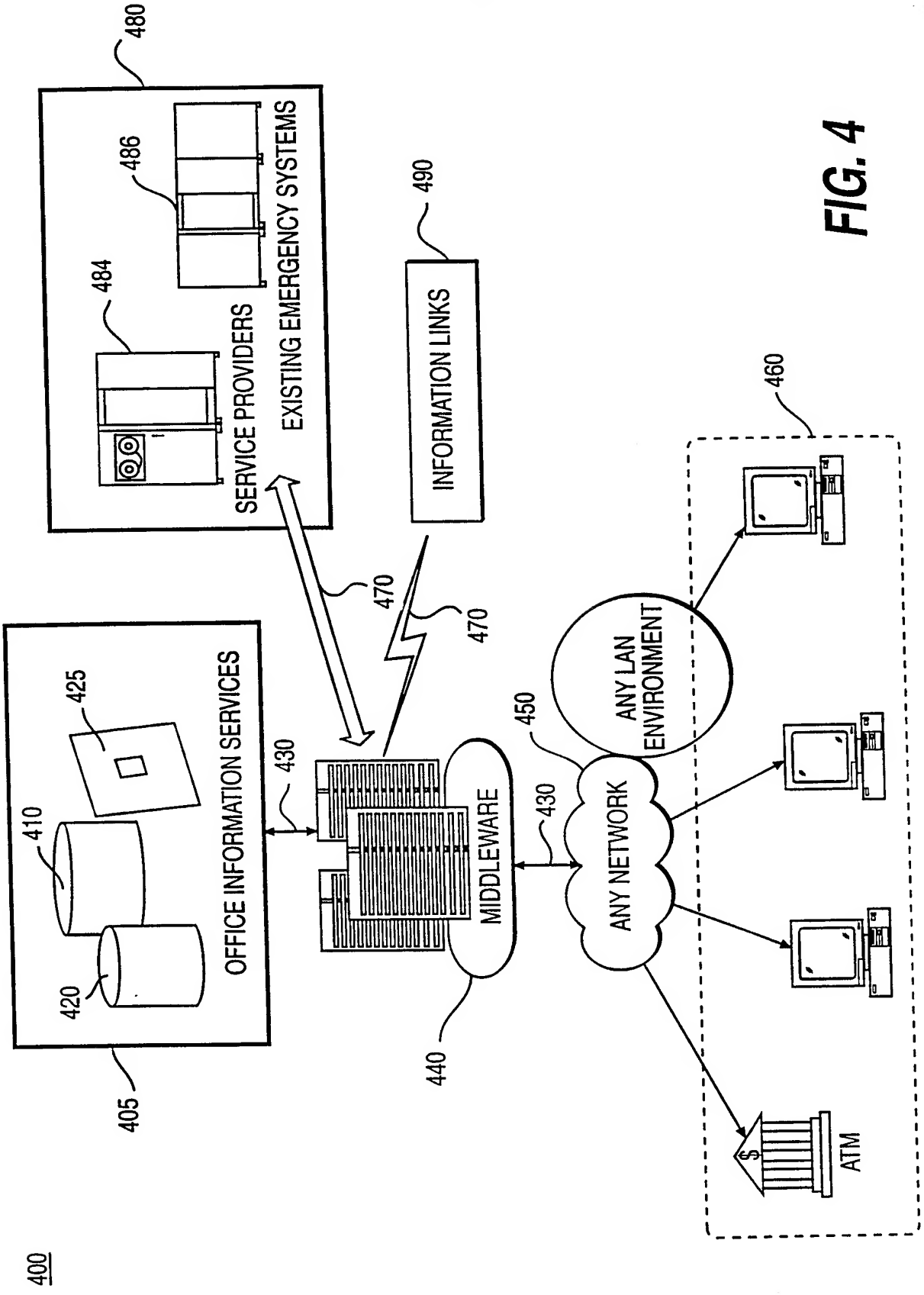

480
SERVICE PROVIDERS
EXISTING EMERGENCY SYSTEMS
486
484
490
INFORMATION LINKS
460
470
470
425
410
420
OFFICE INFORMATION SERVICES
405
430
MIDDLEWARE
440
450
ANY LAN ENVIRONMENT
430
ANY NETWORK
ATM
400

FIG. 4

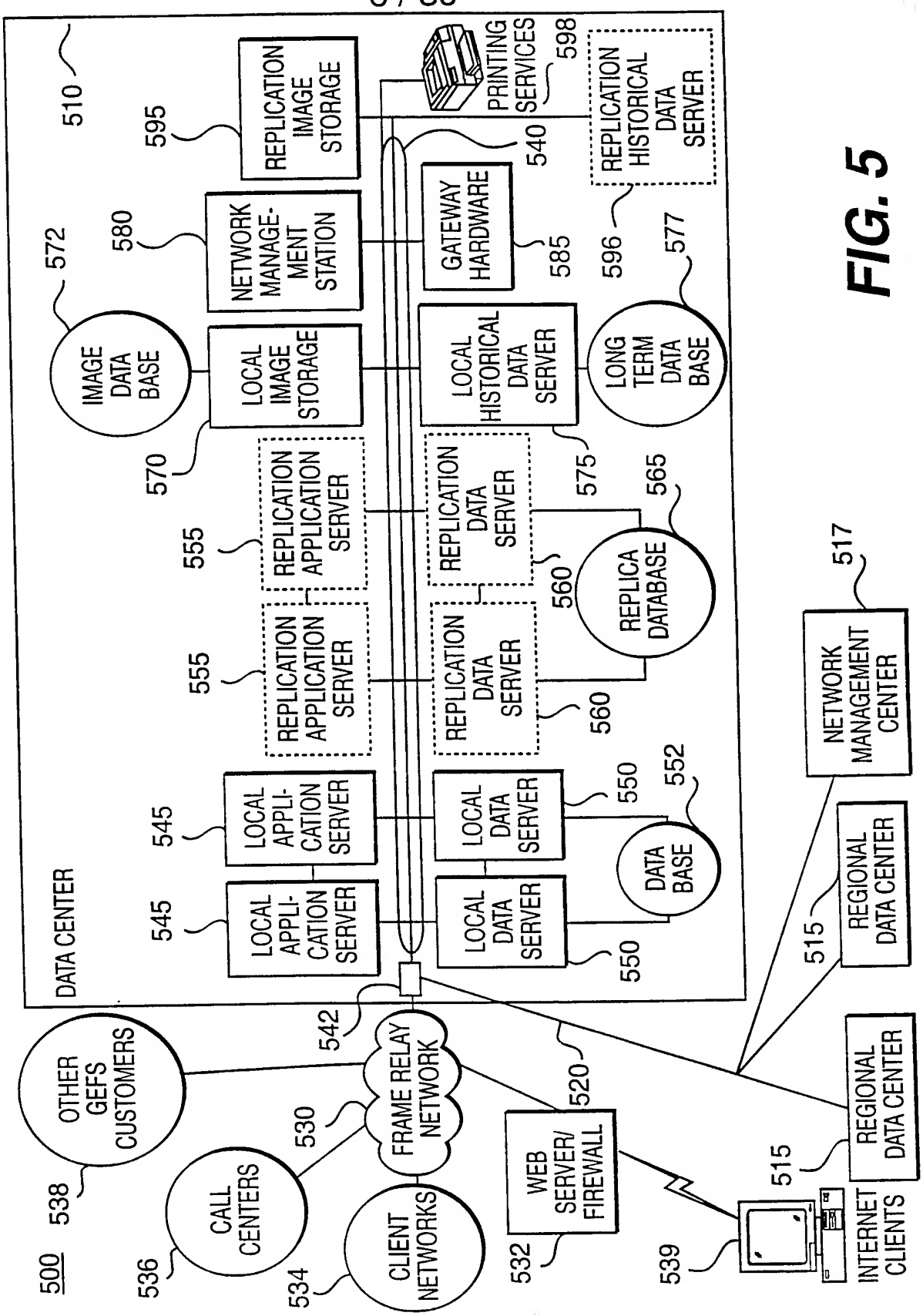

500
INTERNET CLIENTS
539
532
WEB SERVER/ FIREWALL
534
CLIENT NETWORKS
536
CALL CENTERS
538
OTHER GEFS CUSTOMERS
530
FRAME RELAY NETWORK
542
520
DATA CENTER
510
REGIONAL DATA CENTER
515
REGIONAL DATA CENTER
515
NETWORK MANAGEMENT CENTER
517
REPLICA DATABASE
565
REPLICATION DATA SERVER
560
REPLICATION DATA SERVER
560
REPLICATION APPLICATION SERVER
555
REPLICATION APPLICATION SERVER
555
REPLICATION DATA SERVER
575
LOCAL HISTORICAL DATA SERVER
577
LONG TERM DATA BASE
596
REPLICATION HISTORICAL DATA SERVER
598
PRINTING SERVICES
540
GATEWAY HARDWARE
585
NETWORK MANAGE-MENT STATION
580
REPLICATION IMAGE STORAGE
595
LOCAL IMAGE STORAGE
570
IMAGE DATA BASE
572
LOCAL DATA SERVER
550
DATA BASE
552
LOCAL DATA SERVER
550
LOCAL APPLI-CATION SERVER
545
LOCAL APPLI-CATION SERVER
545

FIG. 5

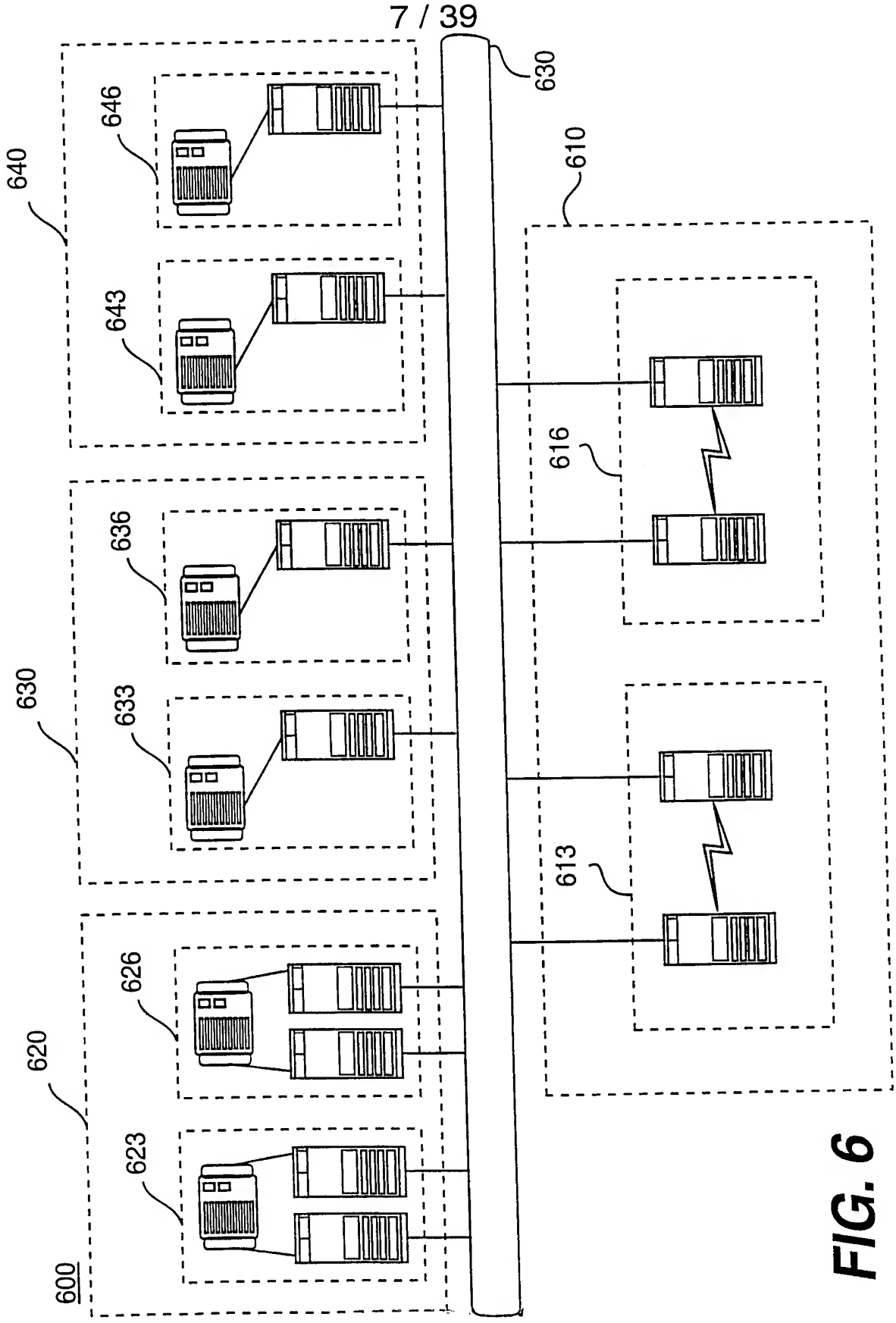


FIG. 6

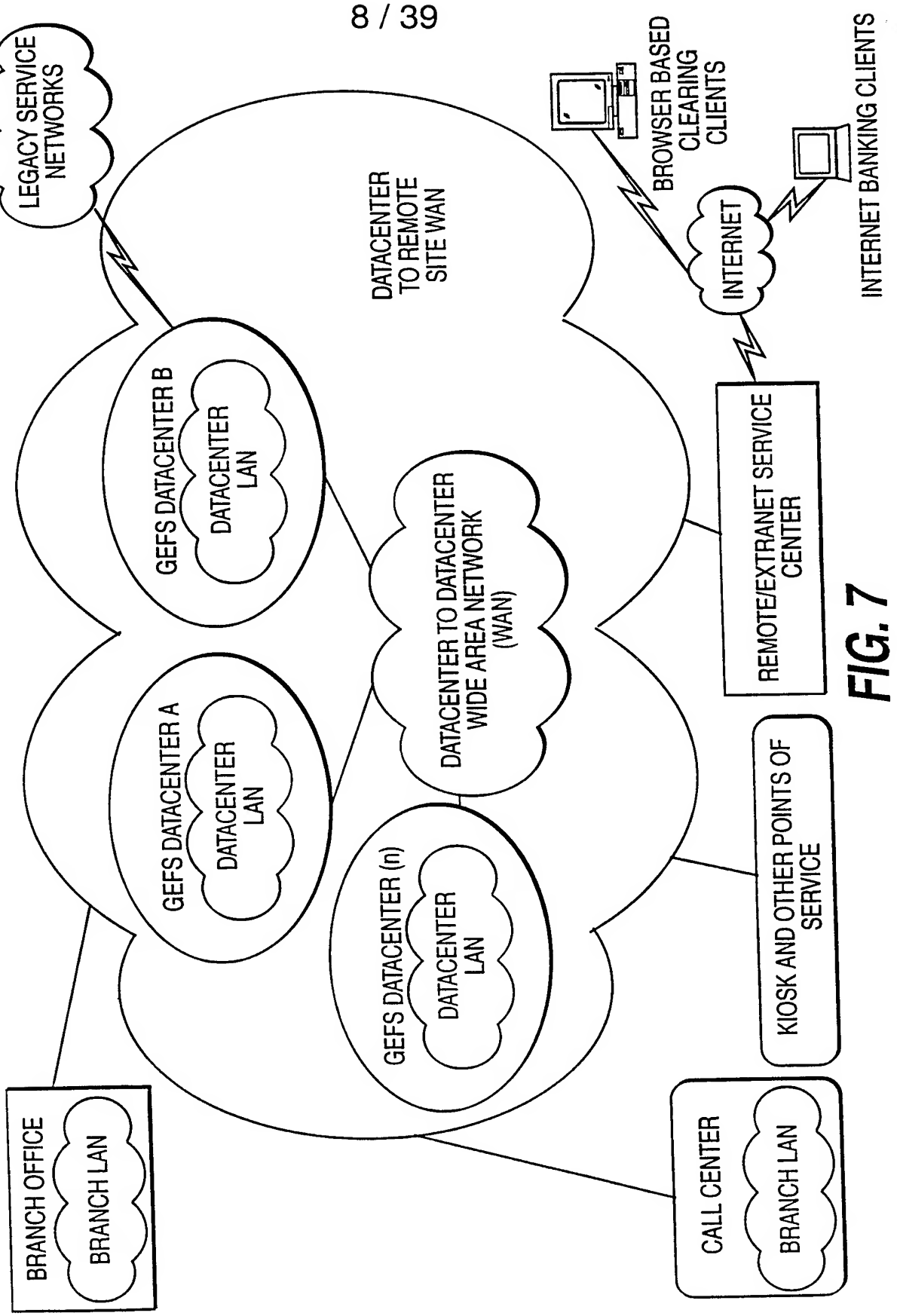

LEGACY SERVICE NETWORKS
DATACENTER TO REMOTE SITE WAN
BROWSER BASED CLEARING CLIENTS
INTERNET
INTERNET BANKING CLIENTS
GEFS DATACENTER B
DATACENTER LAN
DATACENTER TO DATACENTER WIDE AREA NETWORK (WAN)
REMOTE/EXTRANET SERVICE CENTER
GEFS DATACENTER A
DATACENTER LAN
GEFS DATACENTER (n)
DATACENTER LAN
KIOSK AND OTHER POINTS OF SERVICE
BRANCH OFFICE
BRANCH LAN
CALL CENTER
BRANCH LAN

FIG. 7

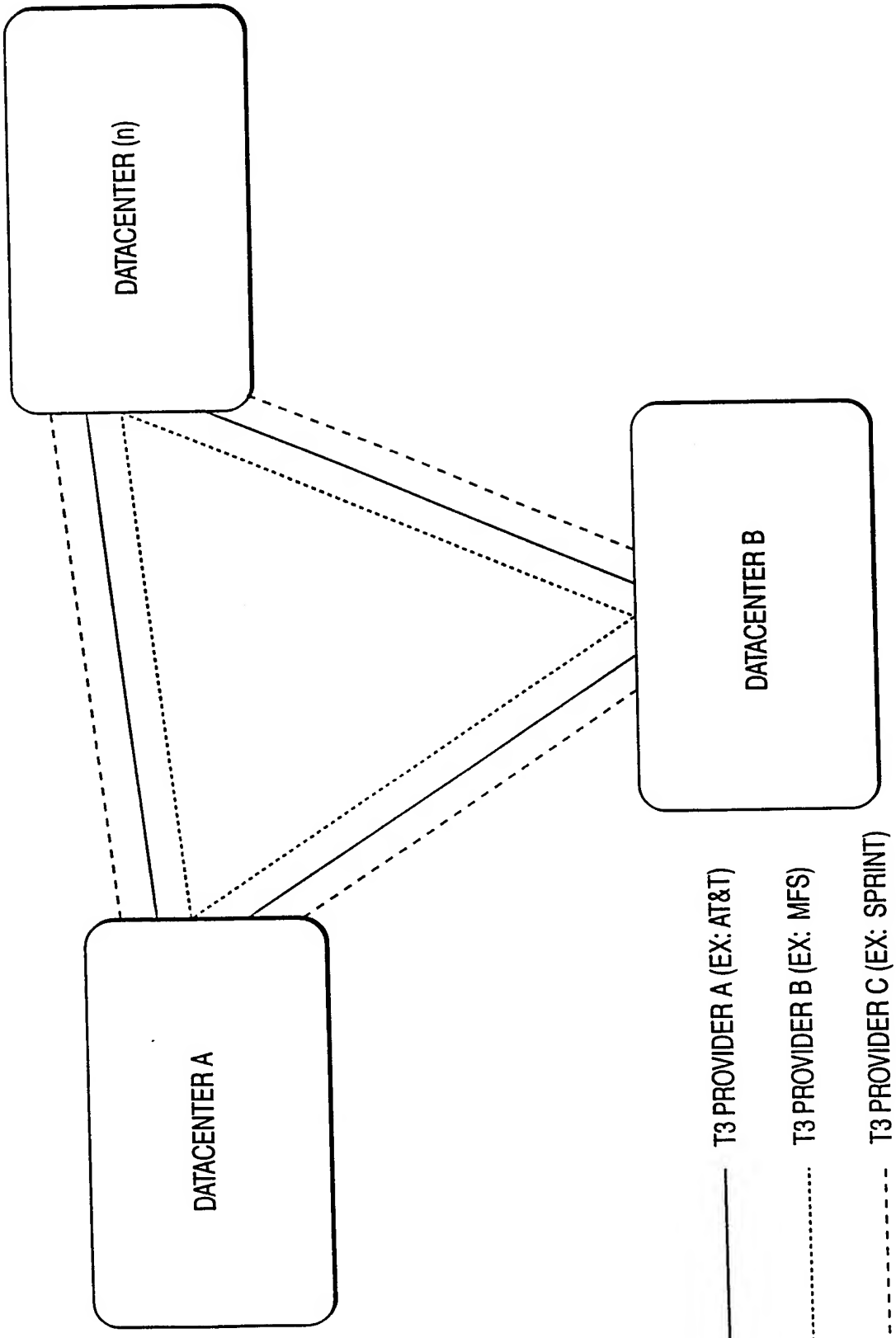

DATACENTER (n)
DATACENTER A
DATACENTER B
T3 PROVIDER A (EX: AT&T)
T3 PROVIDER B (EX: MFS)
T3 PROVIDER C (EX: SPRINT)

FIG. 8

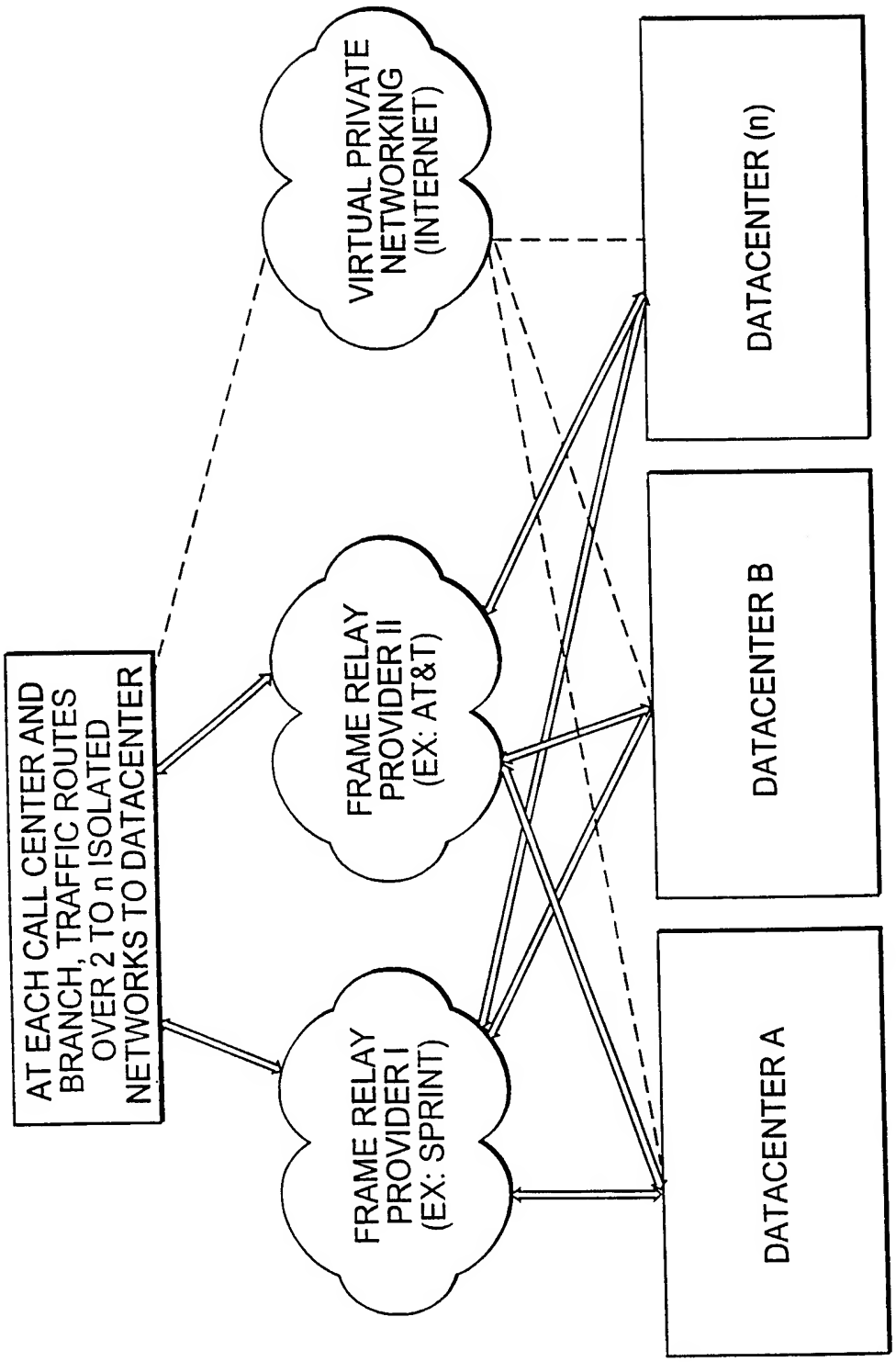

AT EACH CALL CENTER AND BRANCH, TRAFFIC ROUTES OVER 2 TO n ISOLATED NETWORKS TO DATACENTER
FRAME RELAY PROVIDER I (EX: SPRINT)
FRAME RELAY PROVIDER II (EX: AT&T)
VIRTUAL PRIVATE NETWORKING (INTERNET)
DATACENTER A
DATACENTER B
DATACENTER (n)

FIG. 9

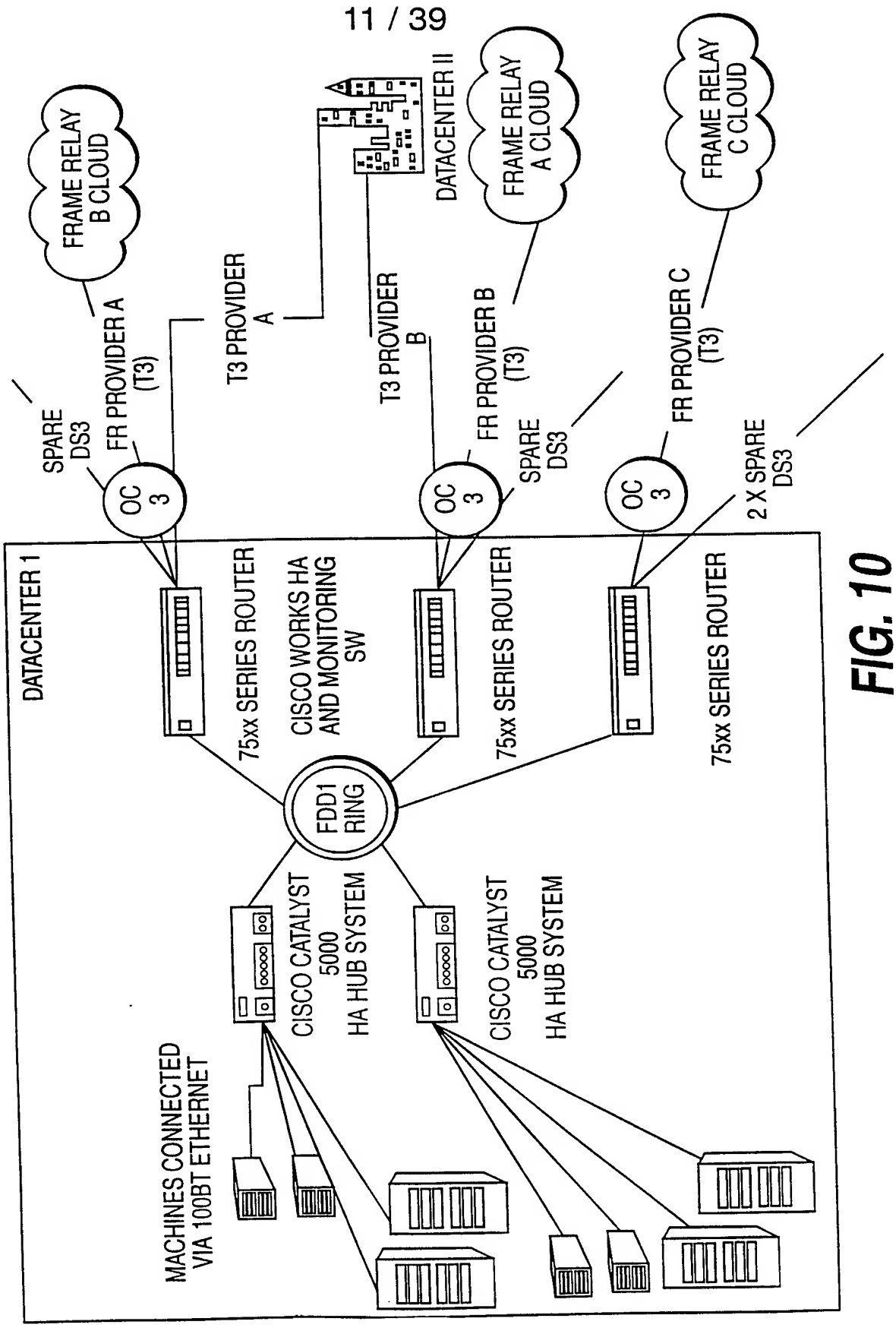

DATACENTER 1
FRAME RELAY B CLOUD
FRAME RELAY A CLOUD
FRAME RELAY C CLOUD
DATACENTER II
SPARE DS3
FR PROVIDER A (T3)
T3 PROVIDER A
T3 PROVIDER B
FR PROVIDER B (T3)
SPARE DS3
FR PROVIDER C (T3)
2 X SPARE DS3
OC 3
OC 3
OC 3
75xx SERIES ROUTER
CISCO WORKS HA AND MONITORING SW
75xx SERIES ROUTER
75xx SERIES ROUTER
FDDI RING
CISCO CATALYST 5000 HA HUB SYSTEM
CISCO CATALYST 5000 HA HUB SYSTEM
MACHINES CONNECTED VIA 100BT ETHERNET

FIG. 10

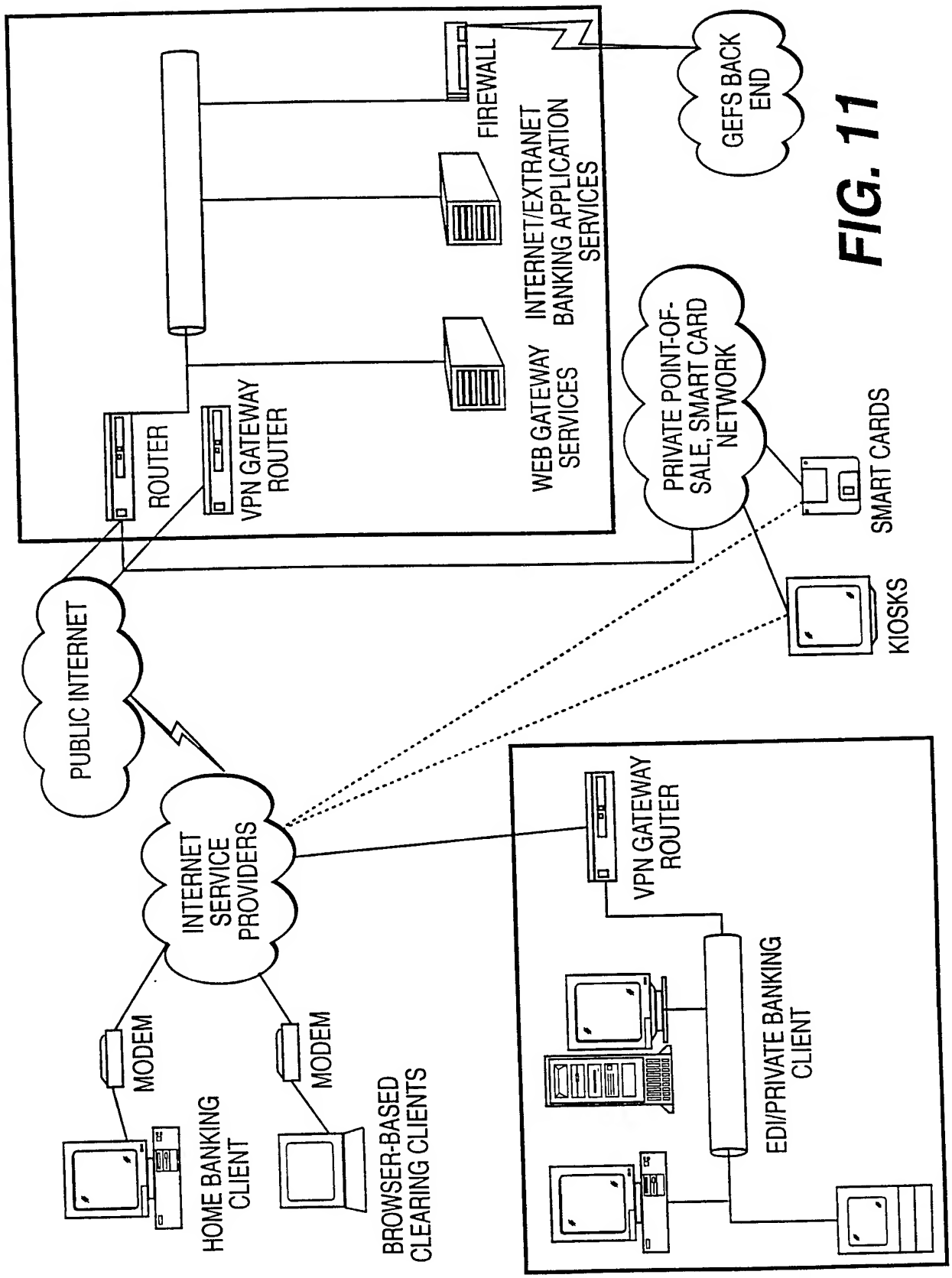

FIREWALL
INTERNET/EXTRANET BANKING APPLICATION SERVICES
GEFS BACK END
WEB GATEWAY SERVICES
ROUTER
VPN GATEWAY ROUTER
PRIVATE POINT-OF-SALE, SMART CARD NETWORK
SMART CARDS
KIOSKS
PUBLIC INTERNET
INTERNET SERVICE PROVIDERS
VPN GATEWAY ROUTER
MODEM
MODEM
HOME BANKING CLIENT
BROWSER-BASED CLEARING CLIENTS
EDI/PRIVATE BANKING CLIENT

FIG. 11

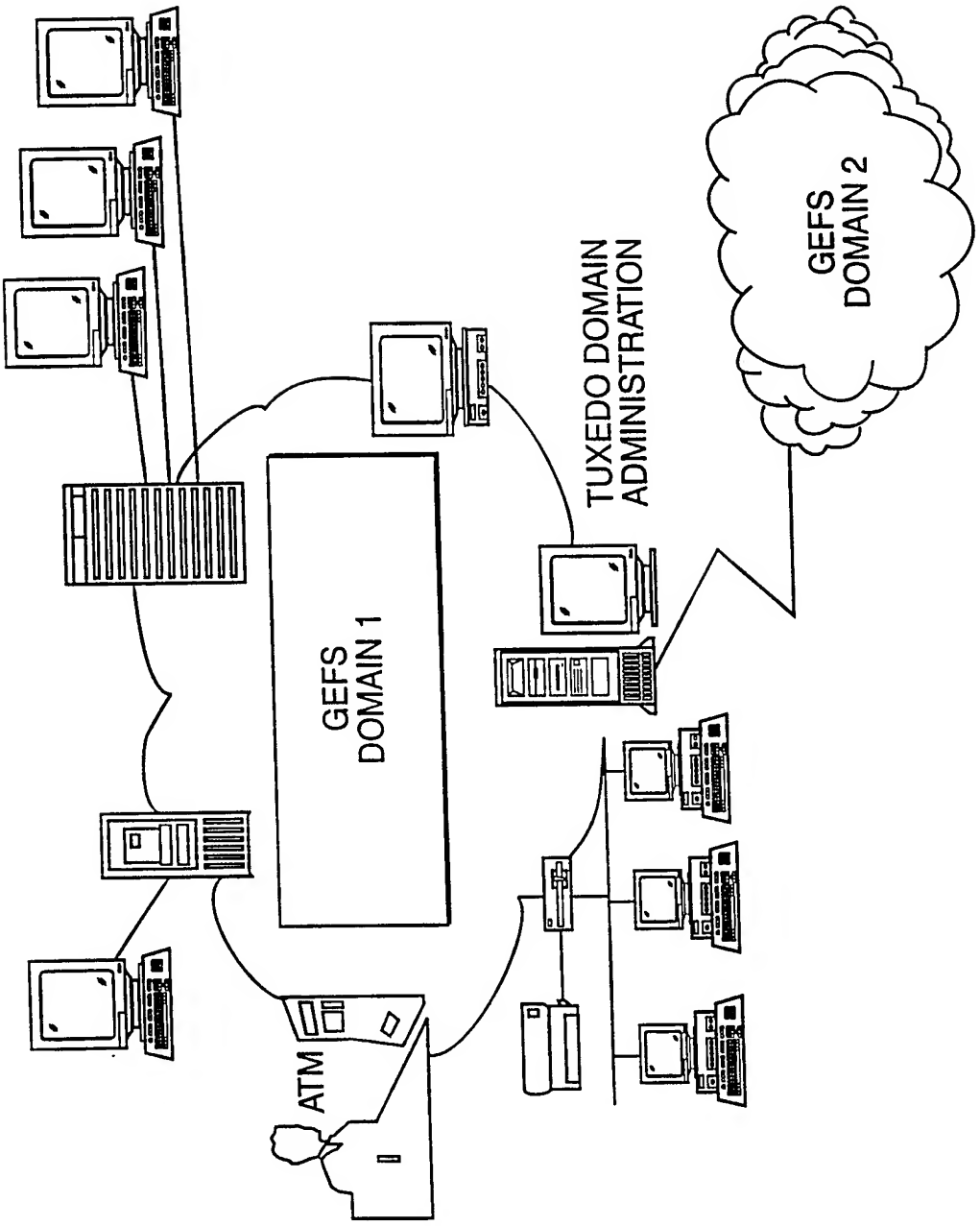

GEFS
DOMAIN 1
TUXEDO DOMAIN
ADMINISTRATION
GEFS
DOMAIN 2
ATM

FIG. 12

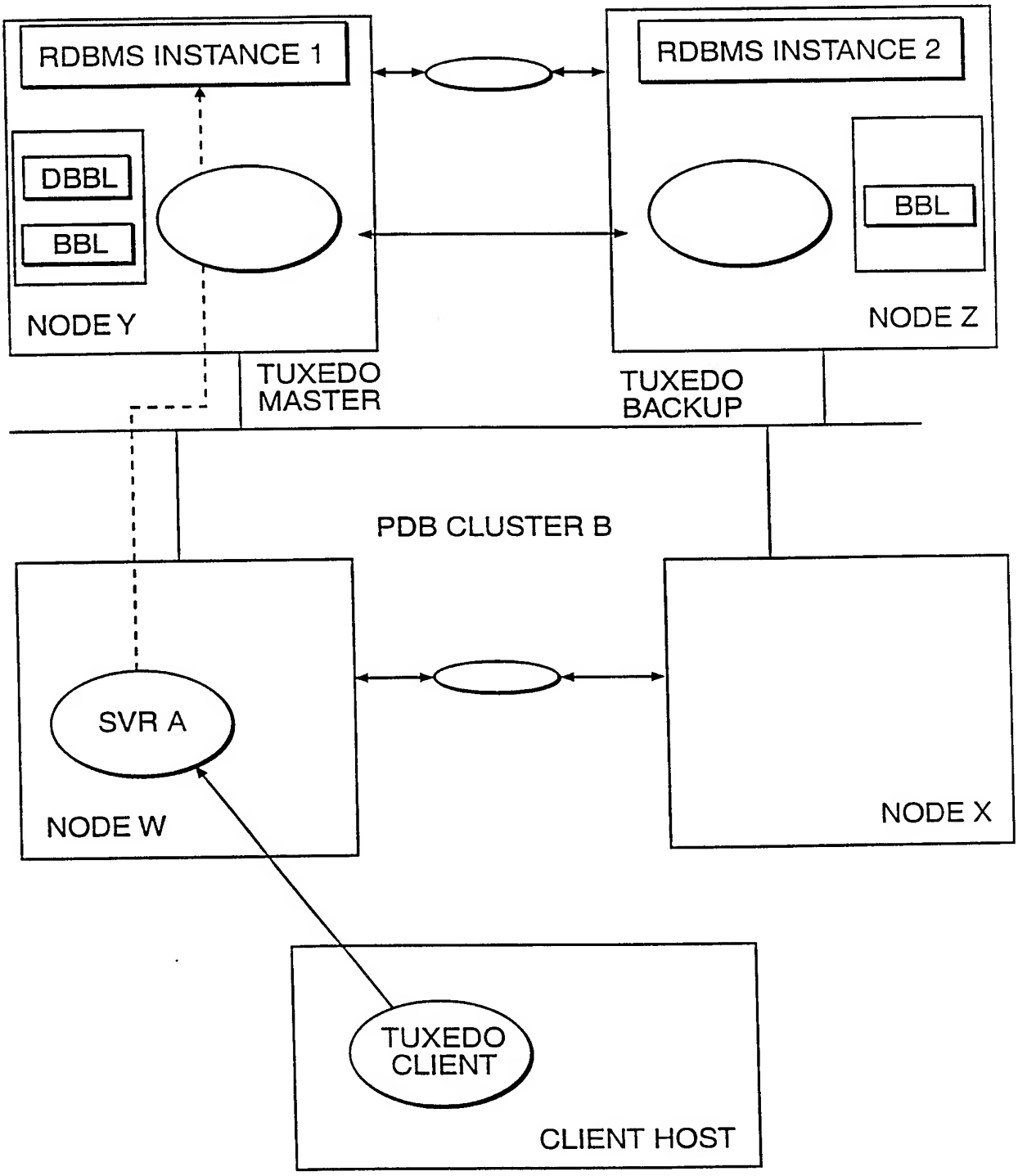

PDB CLUSTER A
RDBMS INSTANCE 1
RDBMS INSTANCE 2
DBBL
BBL
BBL
NODE Y
NODE Z
TUXEDO MASTER
TUXEDO BACKUP
PDB CLUSTER B
SVR A
NODE W
NODE X
TUXEDO CLIENT
CLIENT HOST

FIG.13

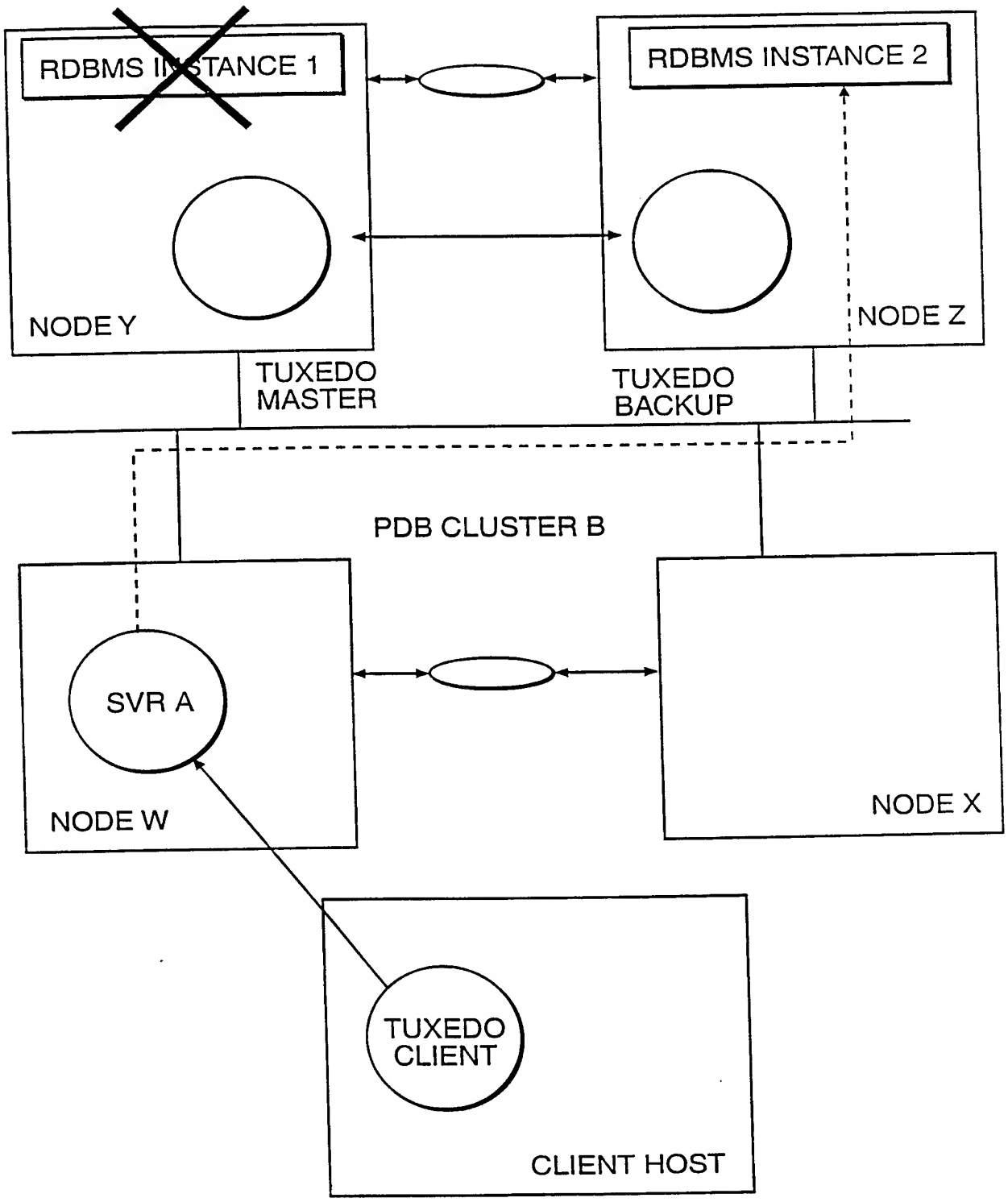

PDB CLUSTER A
RDBMS INSTANCE 1
RDBMS INSTANCE 2
NODE Y
NODE Z
TUXEDO MASTER
TUXEDO BACKUP
PDB CLUSTER B
SVR A
NODE W
NODE X
TUXEDO CLIENT
CLIENT HOST

FIG.14

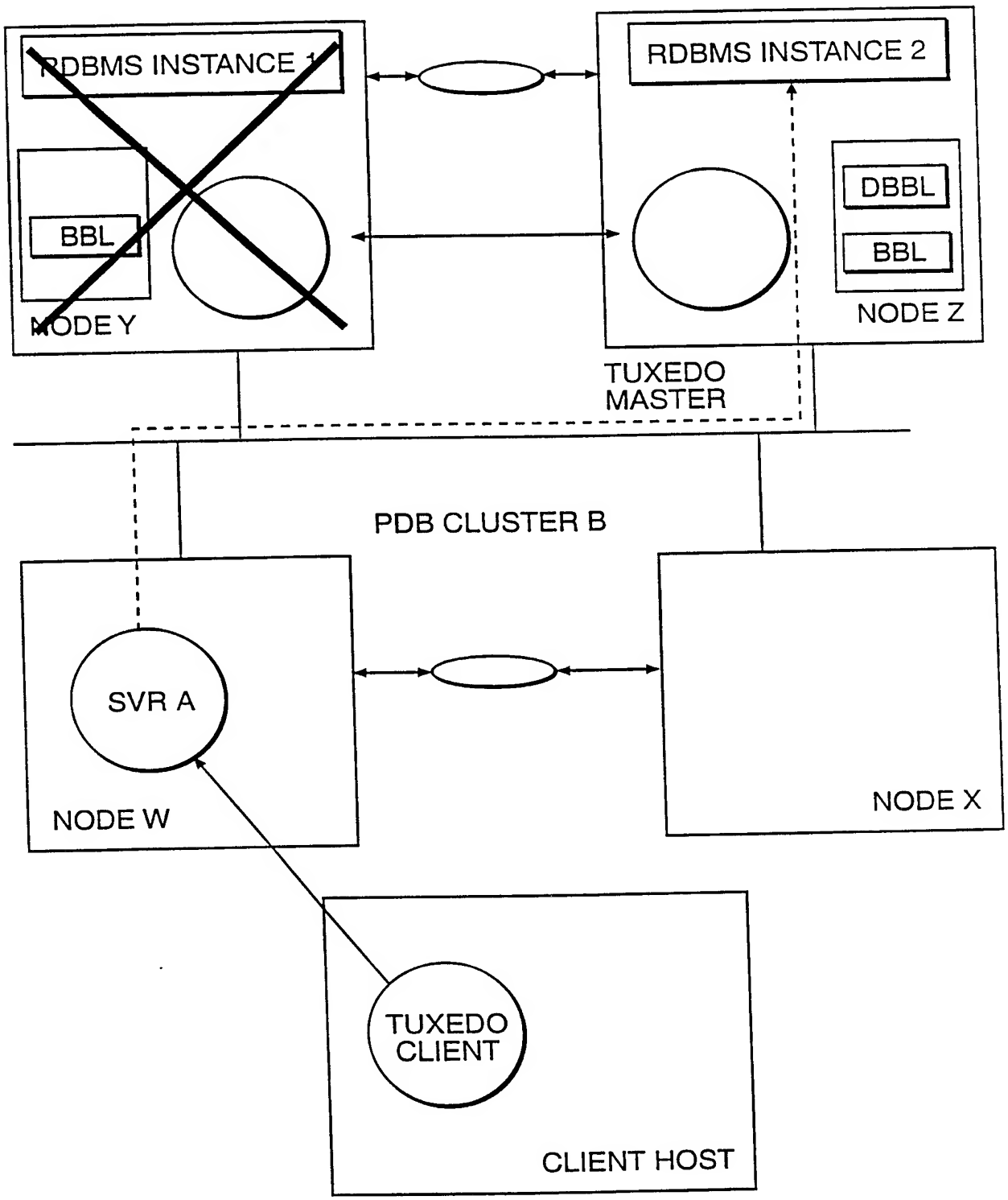

PDB CLUSTER A
RDBMS INSTANCE 1
BBL
NODE Y
RDBMS INSTANCE 2
DBBL
BBL
NODE Z
TUXEDO
MASTER
PDB CLUSTER B
SVR A
NODE W
NODE X
TUXEDO
CLIENT
CLIENT HOST

FIG.15

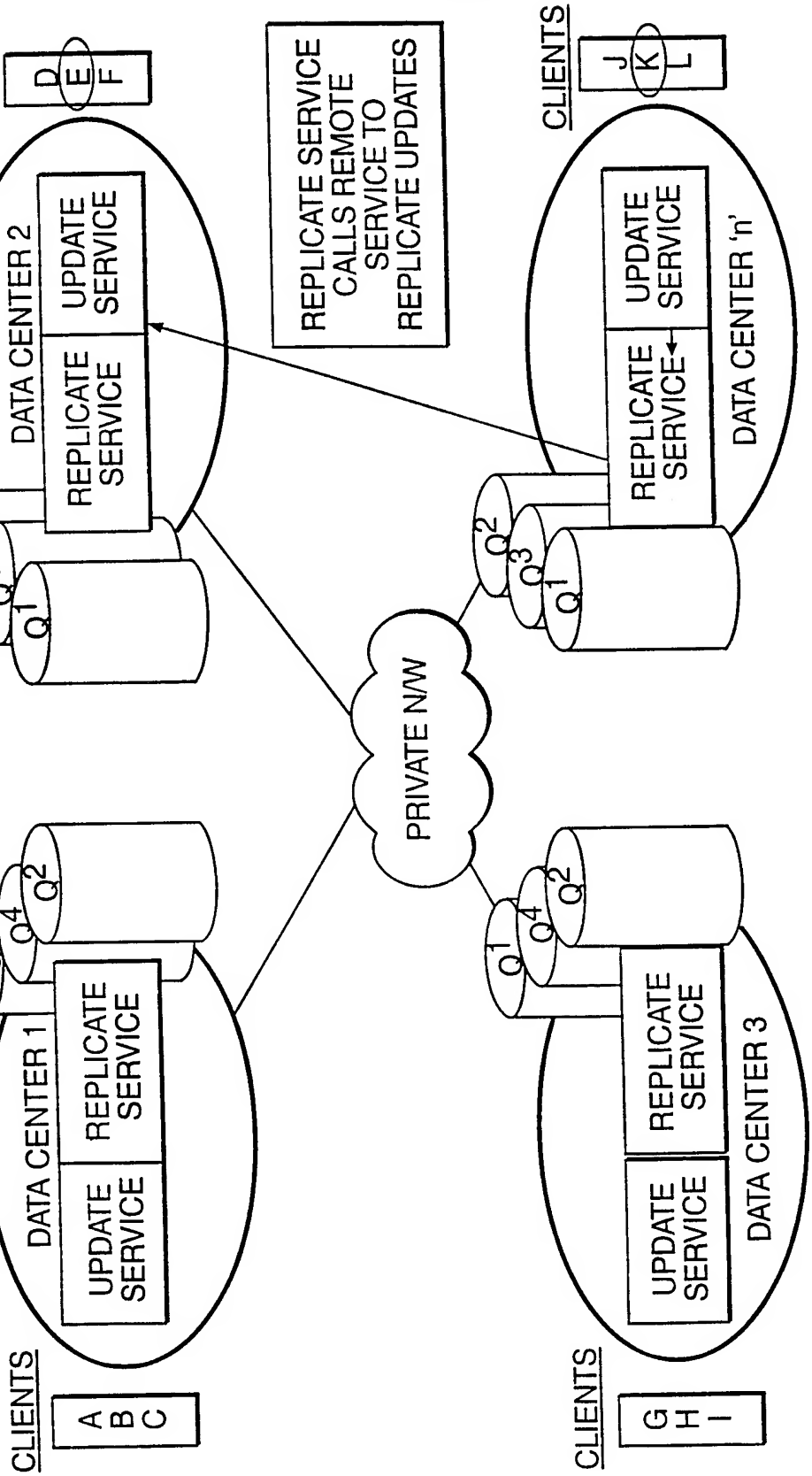

CLIENTS
A
B
C
DATA CENTER 1
UPDATE SERVICE
REPLICATE SERVICE
Q1
Q2
Q3
Q4
CLIENTS
D
E
F
DATA CENTER 2
UPDATE SERVICE
REPLICATE SERVICE
Q1
Q2
Q3
Q4
REPLICATE SERVICE CALLS REMOTE SERVICE TO REPLICATE UPDATES
PRIVATE N/W
CLIENTS
J
K
L
DATA CENTER 'n'
UPDATE SERVICE
REPLICATE SERVICE
Q1
Q2
Q3
CLIENTS
G
H
I
DATA CENTER 3
UPDATE SERVICE
REPLICATE SERVICE
Q1
Q2
Q4

FIG.16A

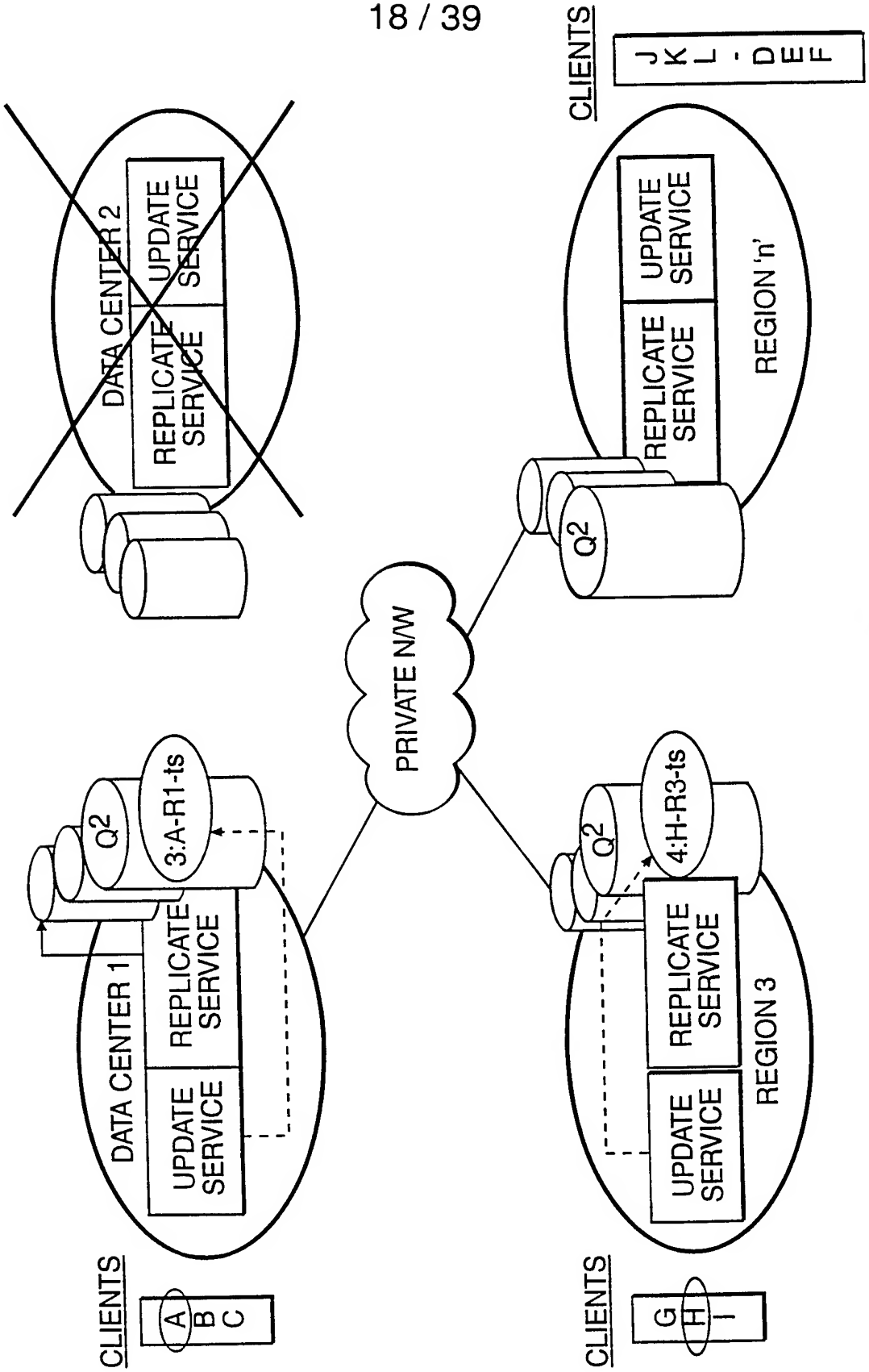

CLIENTS
A
B
C
DATA CENTER 1
UPDATE SERVICE
REPLICATE SERVICE
Q2
3:A-R1-ts
CLIENTS
G
H
I
REGION 3
UPDATE SERVICE
REPLICATE SERVICE
Q2
4:H-R3-ts
PRIVATE N/W
DATA CENTER 2
UPDATE SERVICE
REPLICATE SERVICE
CLIENTS
J
K
L
D
E
F
REGION 'n'
UPDATE SERVICE
REPLICATE SERVICE
Q2

FIG.16B

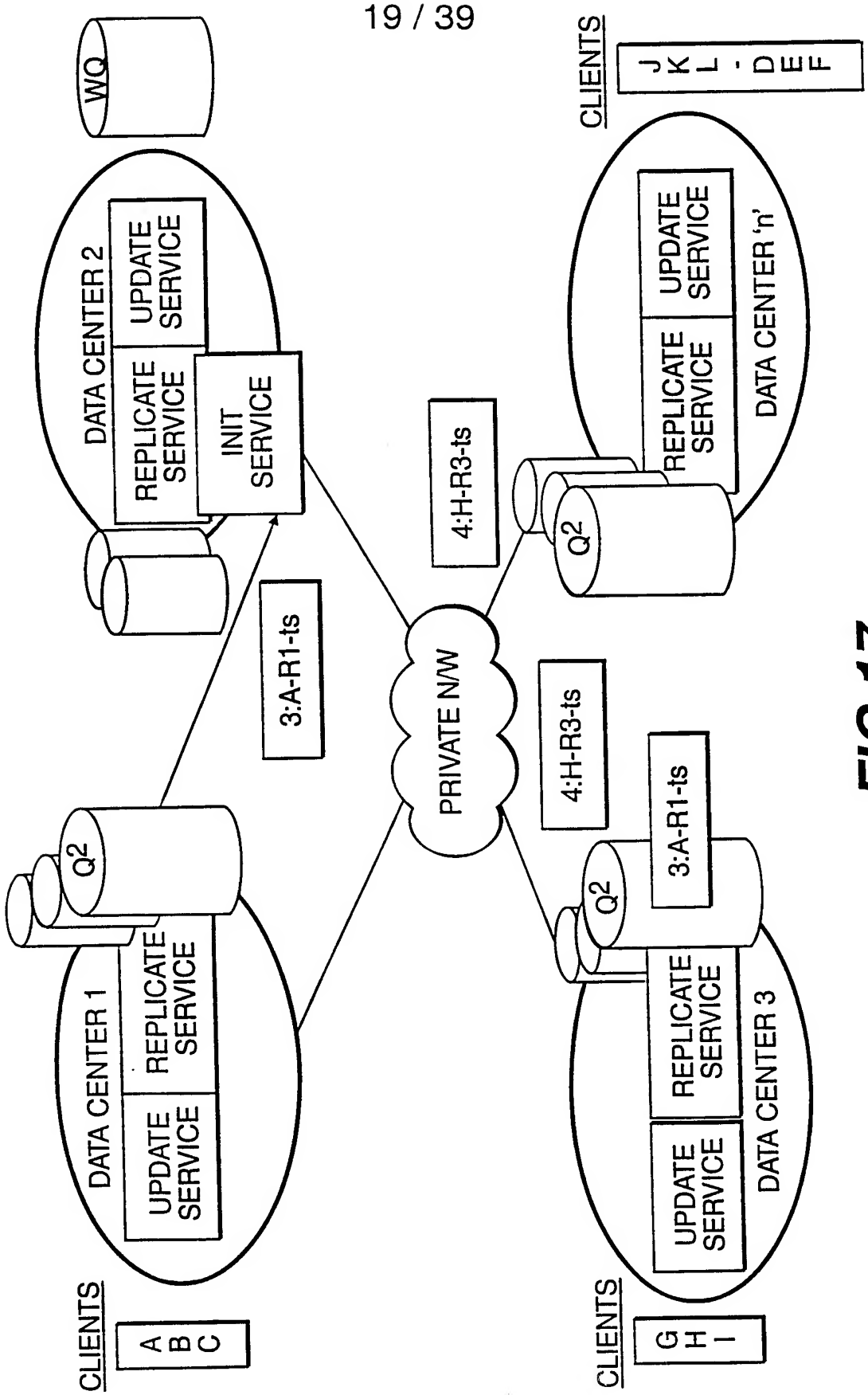

WQ
DATA CENTER 2
UPDATE SERVICE
REPLICATE SERVICE
INIT SERVICE
3:A-R1-ts
4:H-R3-ts
PRIVATE N/W
Q2
DATA CENTER 1
UPDATE SERVICE
REPLICATE SERVICE
CLIENTS
A
B
C
DATA CENTER 'n'
UPDATE SERVICE
REPLICATE SERVICE
CLIENTS
J
K
L
-
D
E
F
DATA CENTER 3
UPDATE SERVICE
REPLICATE SERVICE
CLIENTS
G
H
I

FIG.17

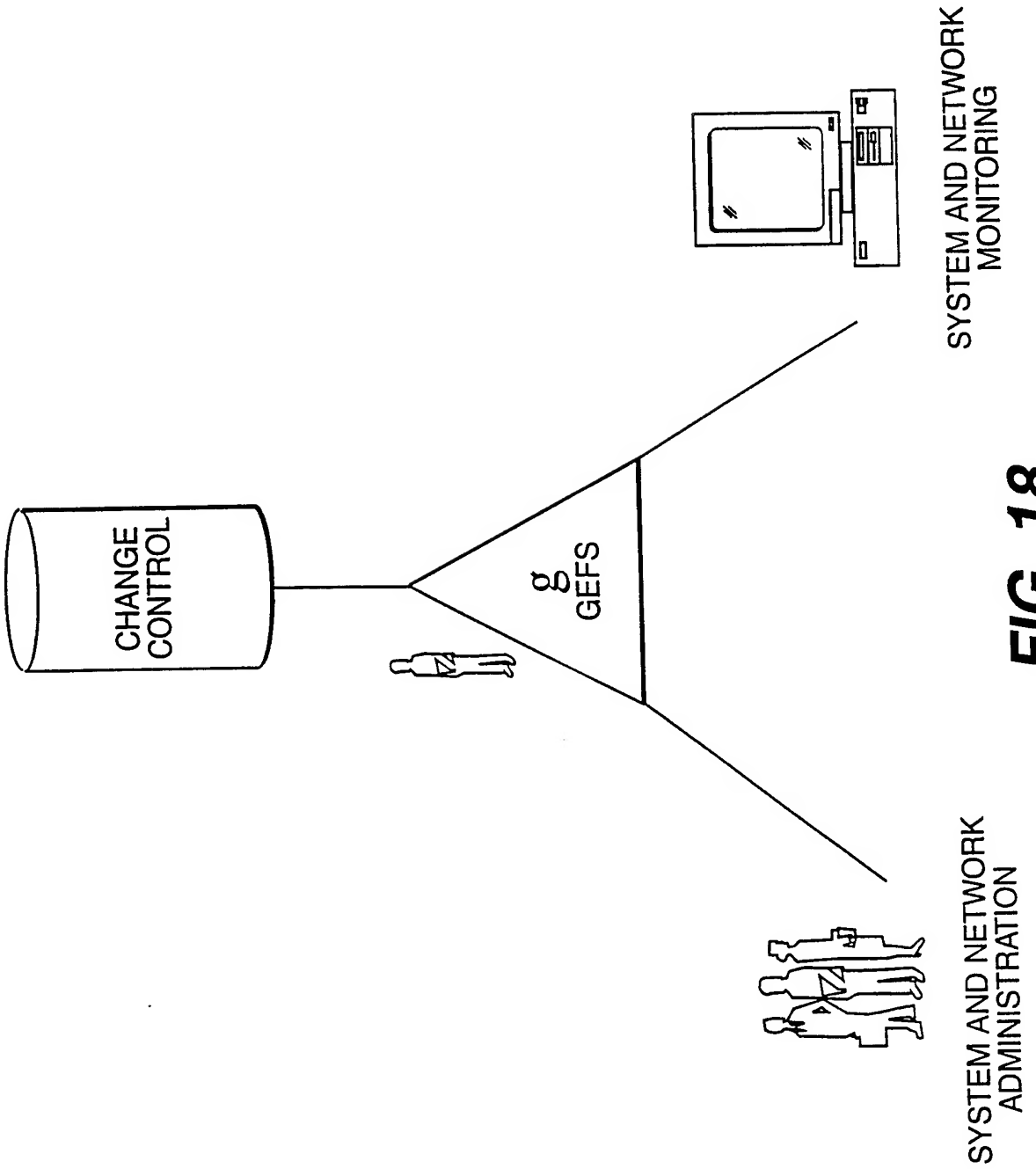

CHANGE CONTROL
GEFS
SYSTEM AND NETWORK MONITORING
SYSTEM AND NETWORK ADMINISTRATION

FIG. 18

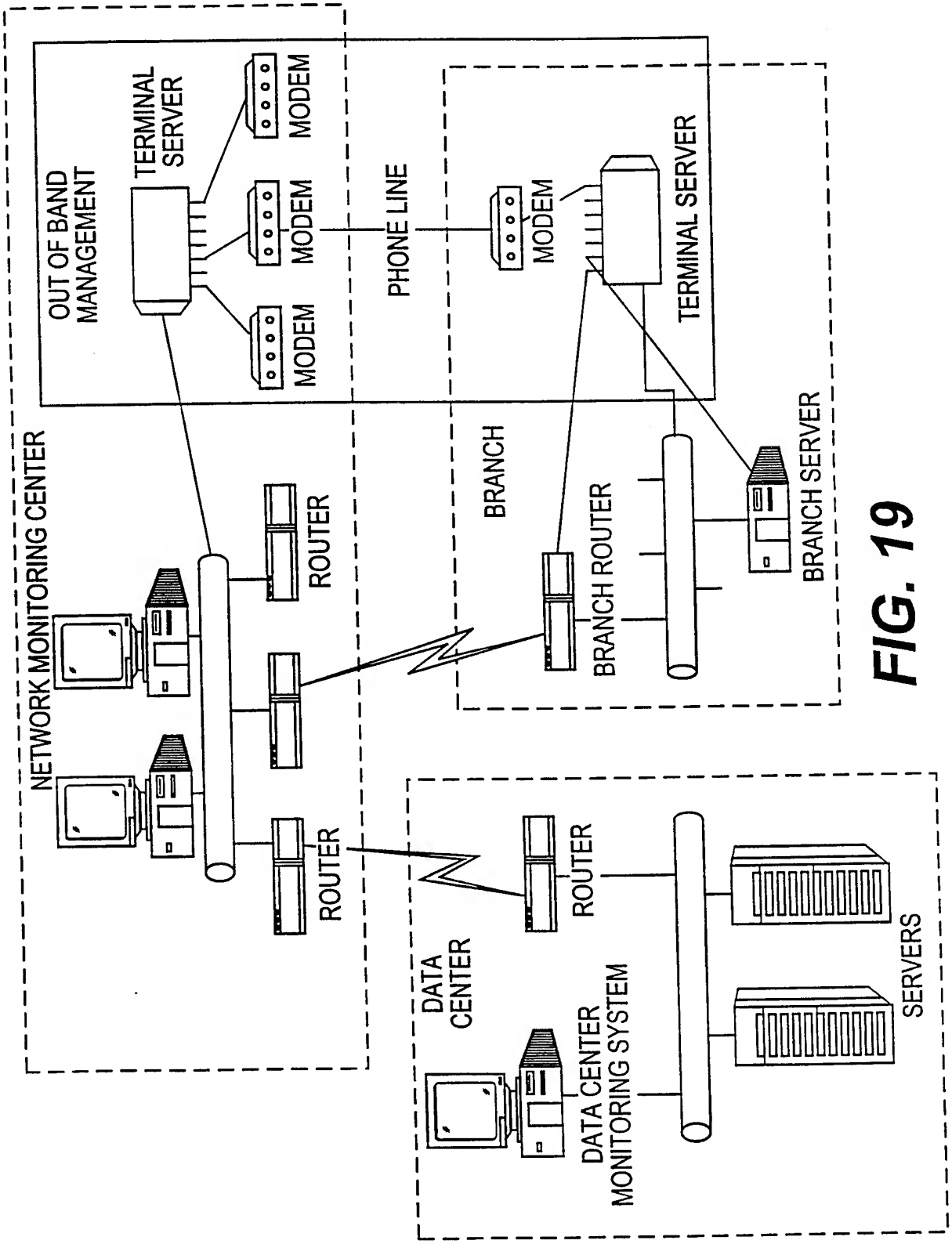

NETWORK MONITORING CENTER
OUT OF BAND MANAGEMENT
TERMINAL SERVER
MODEM
MODEM
MODEM
PHONE LINE
MODEM
TERMINAL SERVER
BRANCH
BRANCH ROUTER
BRANCH SERVER
ROUTER
ROUTER
DATA CENTER
ROUTER
DATA CENTER MONITORING SYSTEM
SERVERS

FIG. 19

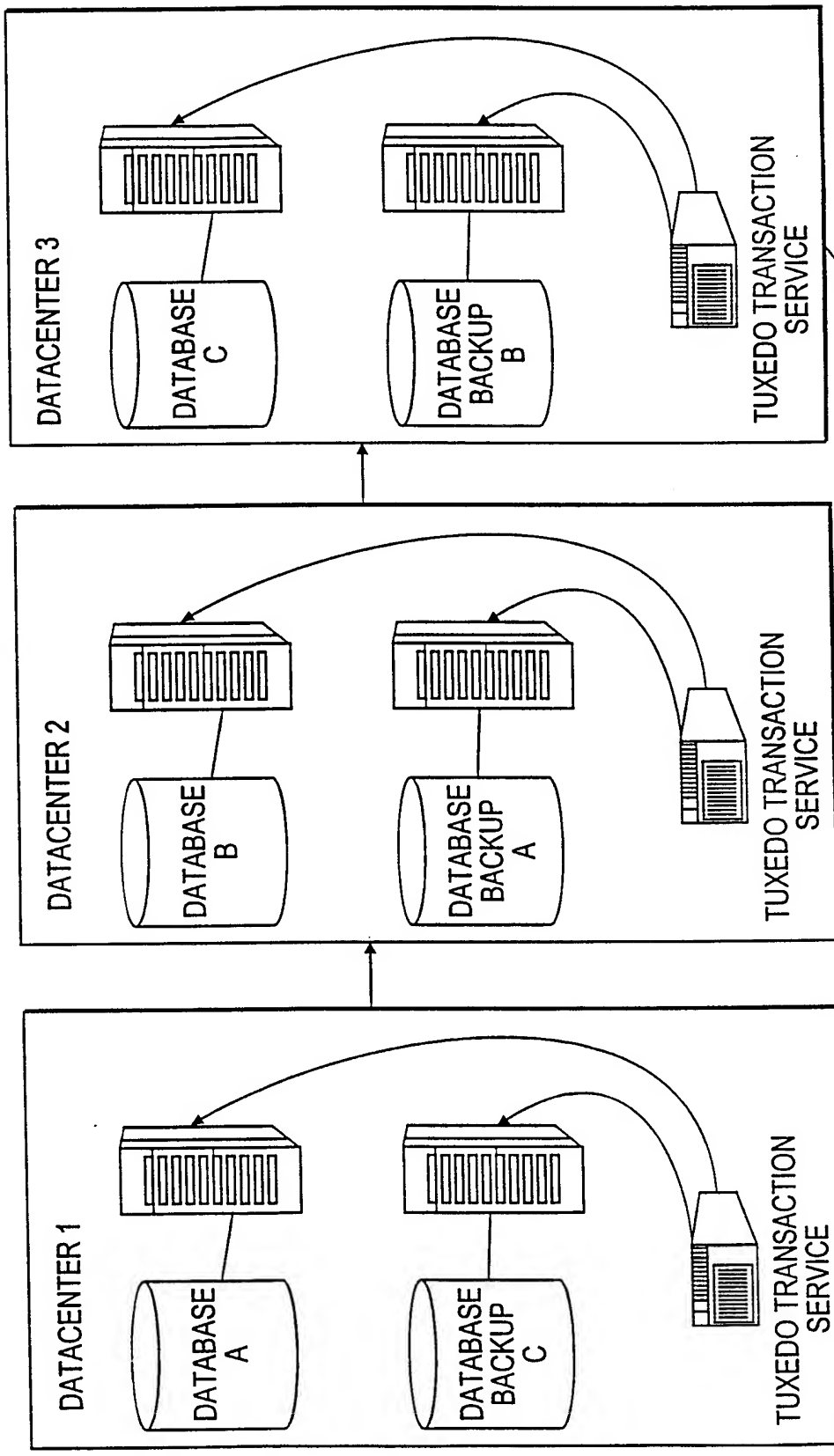

DATACENTER 3
DATABASE C
DATABASE BACKUP B
TUXEDO TRANSACTION SERVICE
DATACENTER 2
DATABASE B
DATABASE BACKUP A
TUXEDO TRANSACTION SERVICE
DATACENTER 1
DATABASE A
DATABASE BACKUP C
TUXEDO TRANSACTION SERVICE

FIG. 20

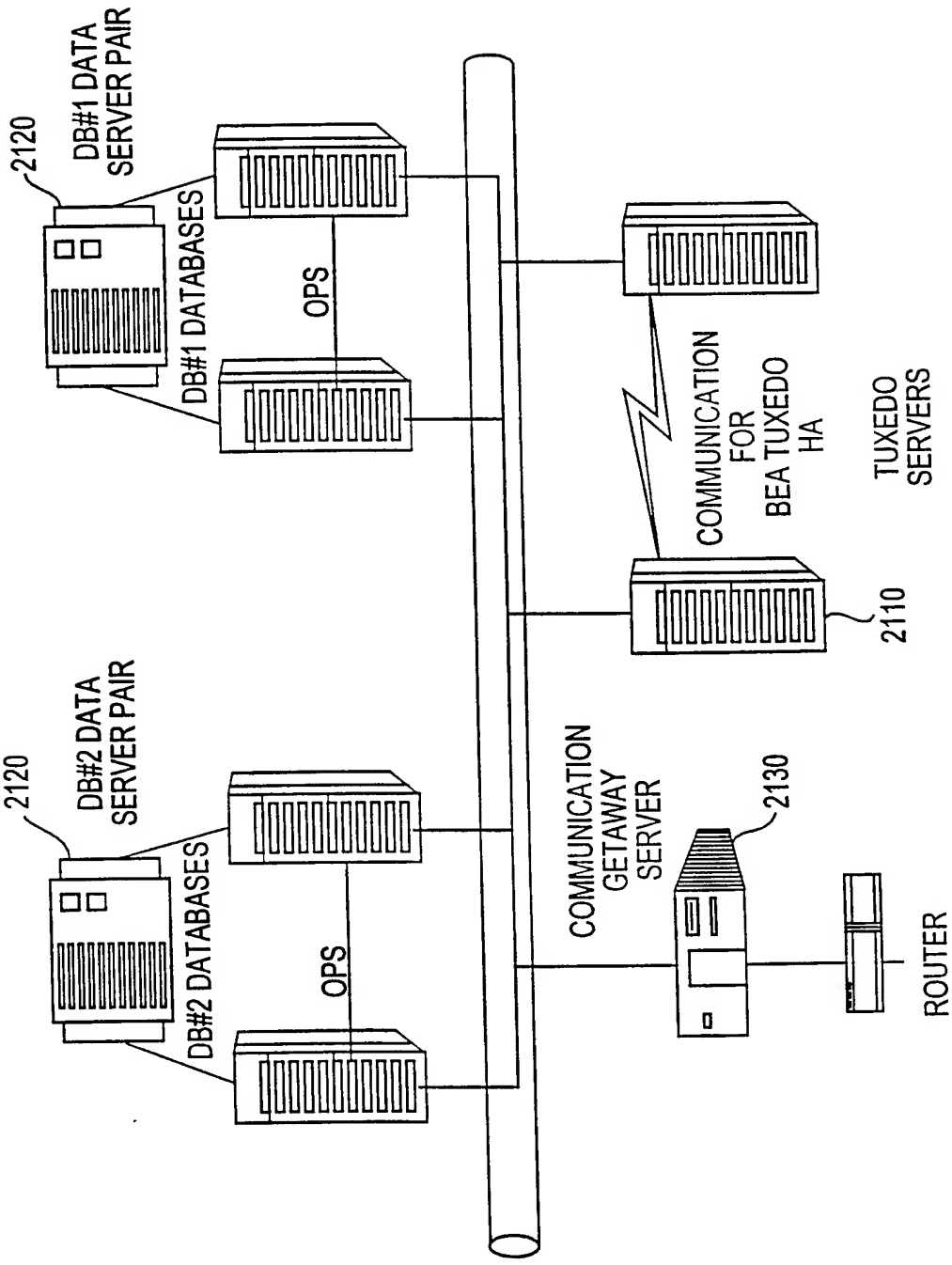

DB#1 DATA SERVER PAIR
2120
DB#1 DATABASES
OPS
COMMUNICATION FOR BEA TUXEDO HA
TUXEDO SERVERS
2110
DB#2 DATA SERVER PAIR
2120
DB#2 DATABASES
OPS
COMMUNICATION GETAWAY SERVER
2130
ROUTER

FIG. 21

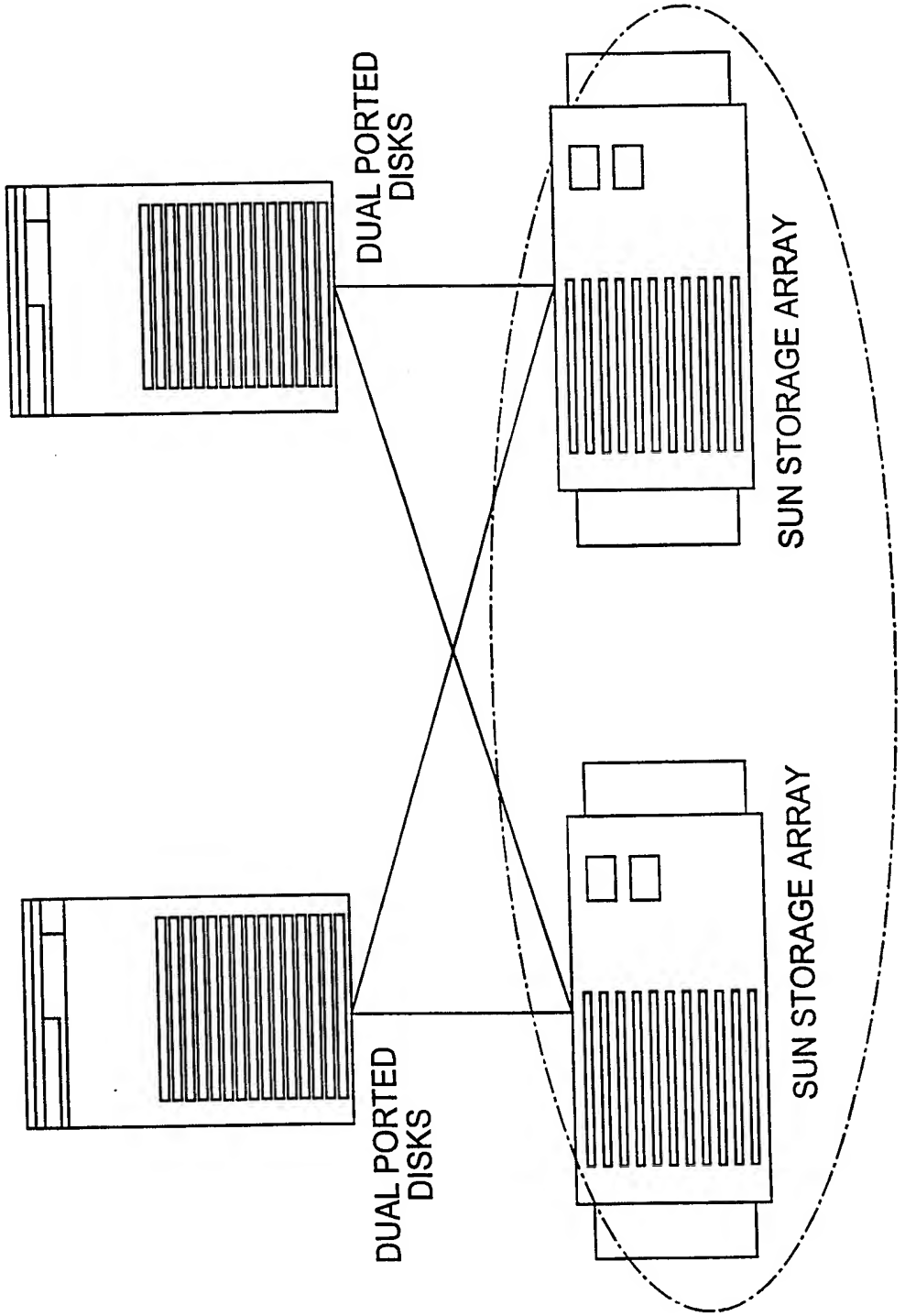

DUAL PORTED DISKS
SUN STORAGE ARRAY
ONE ORACLE DATABASE
SUN STORAGE ARRAY
DUAL PORTED DISKS

FIG. 22

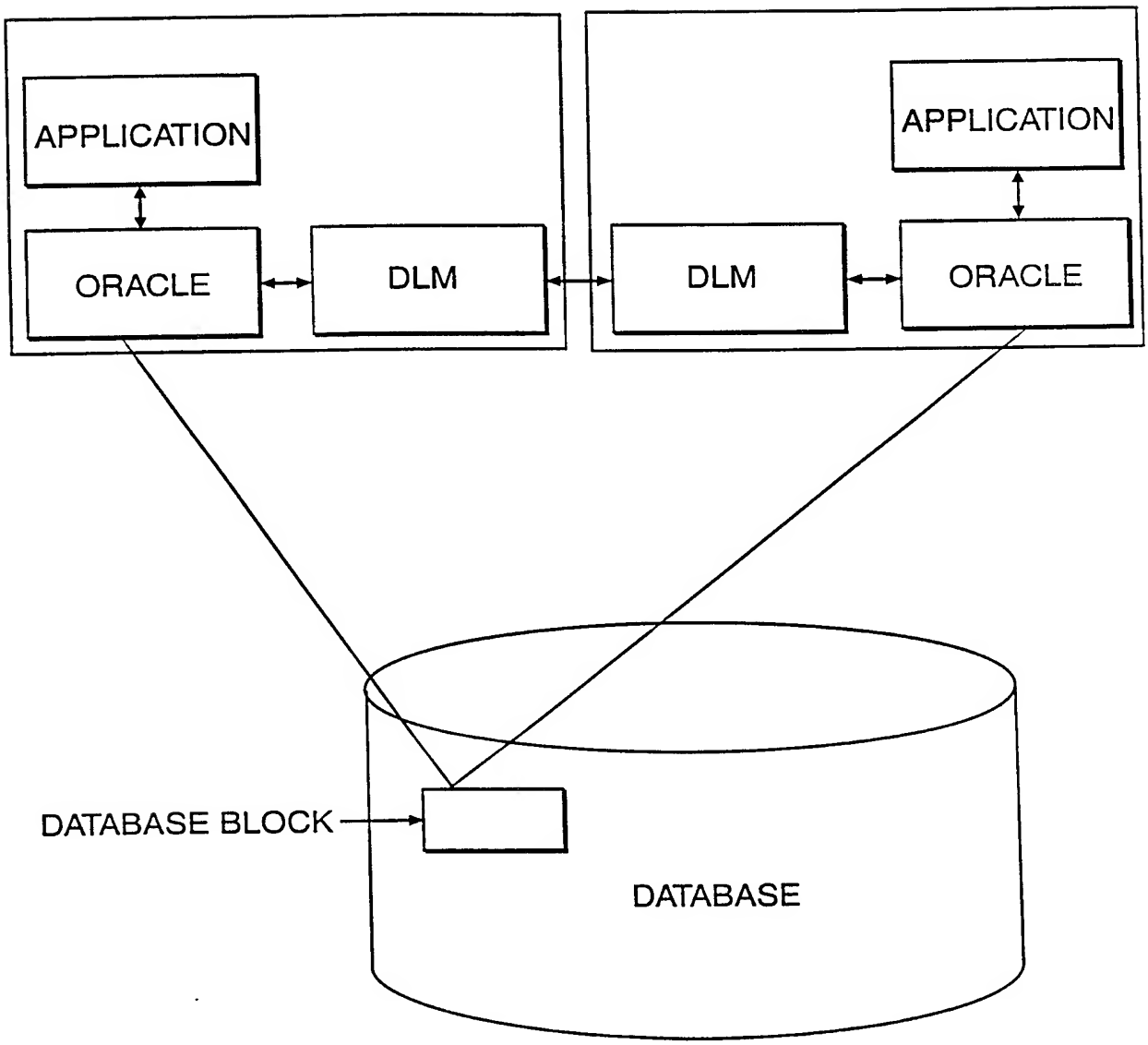

APPLICATION
ORACLE
DLM
DLM
APPLICATION
ORACLE
DATABASE BLOCK
DATABASE

FIG. 23

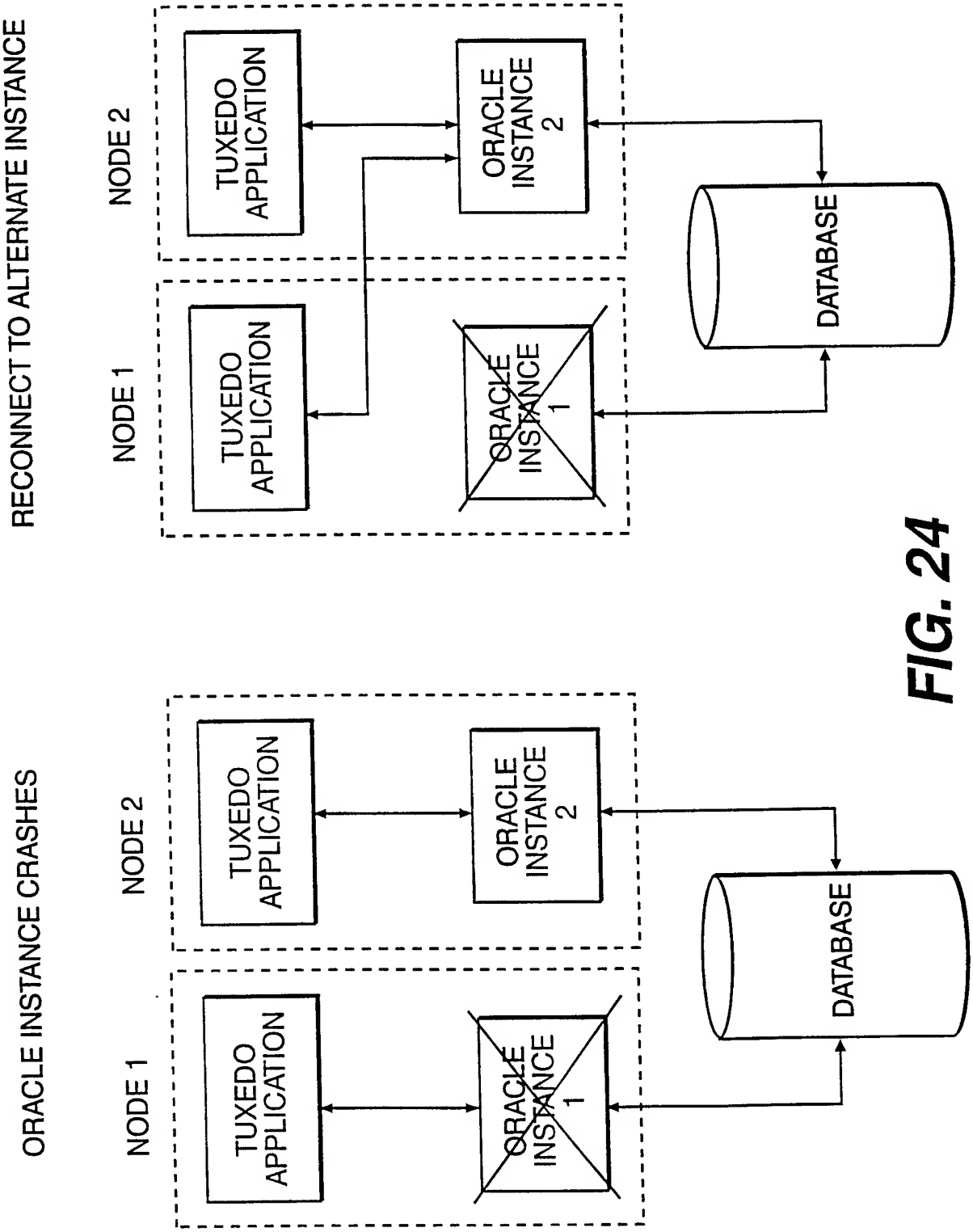

RECONNECT TO ALTERNATE INSTANCE
NODE 2
TUXEDO APPLICATION
ORACLE INSTANCE 2
NODE 1
TUXEDO APPLICATION
ORACLE INSTANCE 1
DATABASE
ORACLE INSTANCE CRASHES
NODE 2
TUXEDO APPLICATION
ORACLE INSTANCE 2
NODE 1
TUXEDO APPLICATION
ORACLE INSTANCE 1
DATABASE

FIG. 24

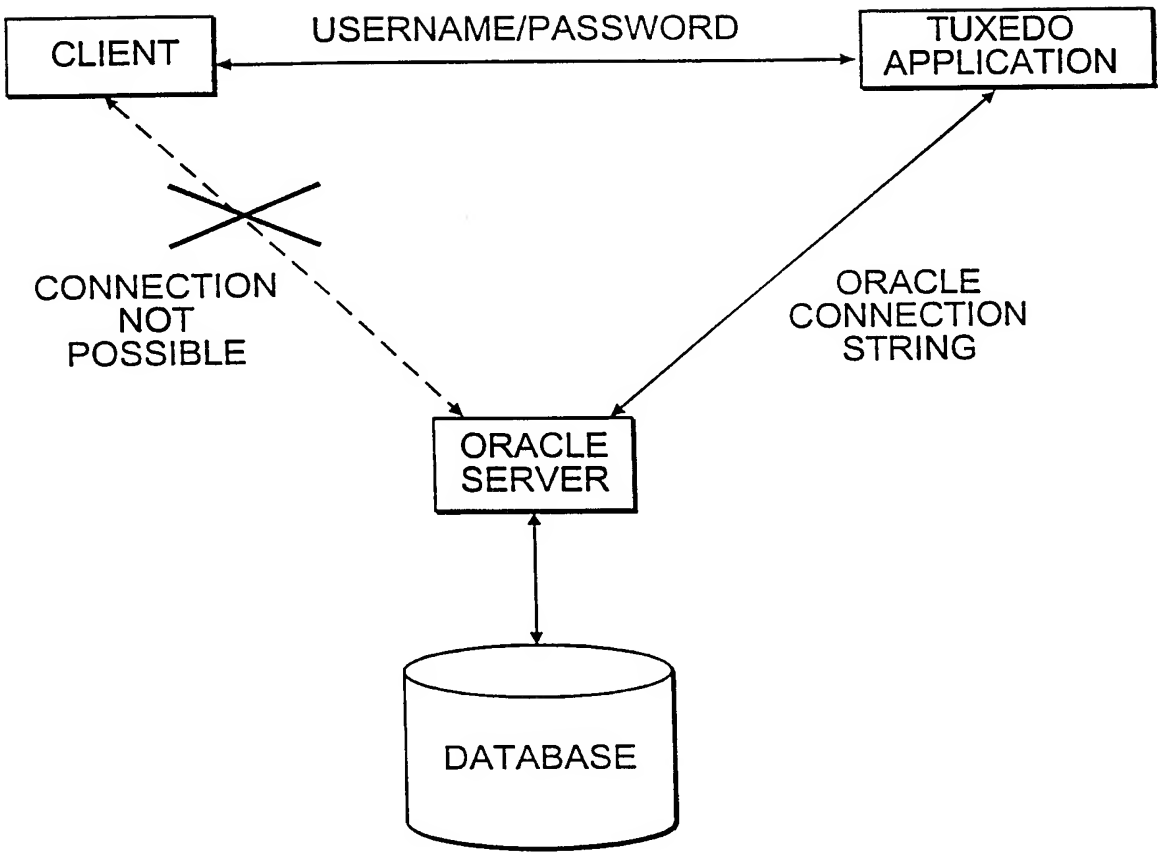

CLIENT
USERNAME/PASSWORD
TUXEDO APPLICATION
CONNECTION NOT POSSIBLE
ORACLE CONNECTION STRING
ORACLE SERVER
DATABASE

FIG. 25

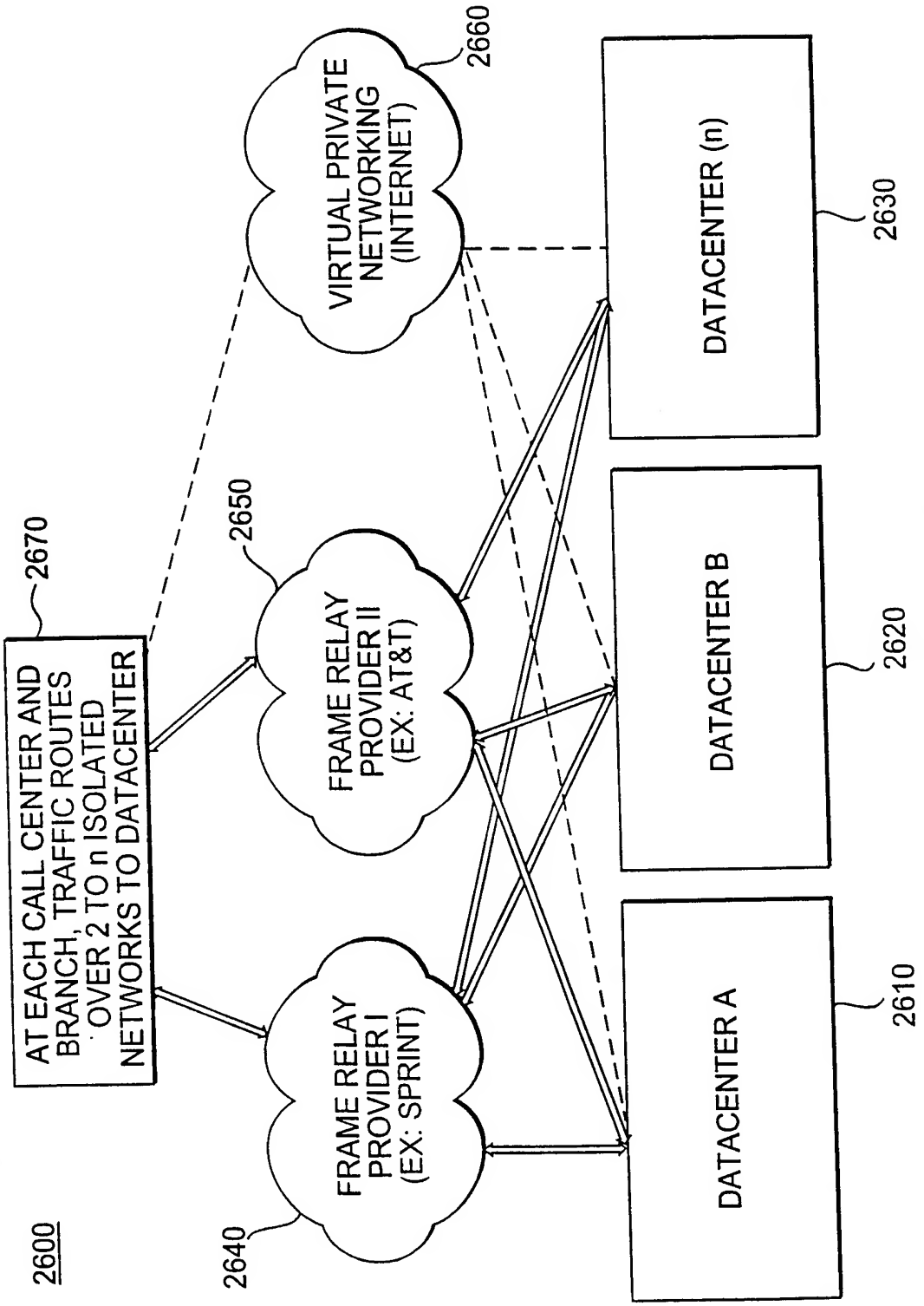

2600
AT EACH CALL CENTER AND BRANCH, TRAFFIC ROUTES OVER 2 TO n ISOLATED NETWORKS TO DATACENTER
2670
FRAME RELAY PROVIDER I (EX: SPRINT)
2640
FRAME RELAY PROVIDER II (EX: AT&T)
2650
VIRTUAL PRIVATE NETWORKING (INTERNET)
2660
DATACENTER A
2610
DATACENTER B
2620
DATACENTER (n)
2630

FIG. 26

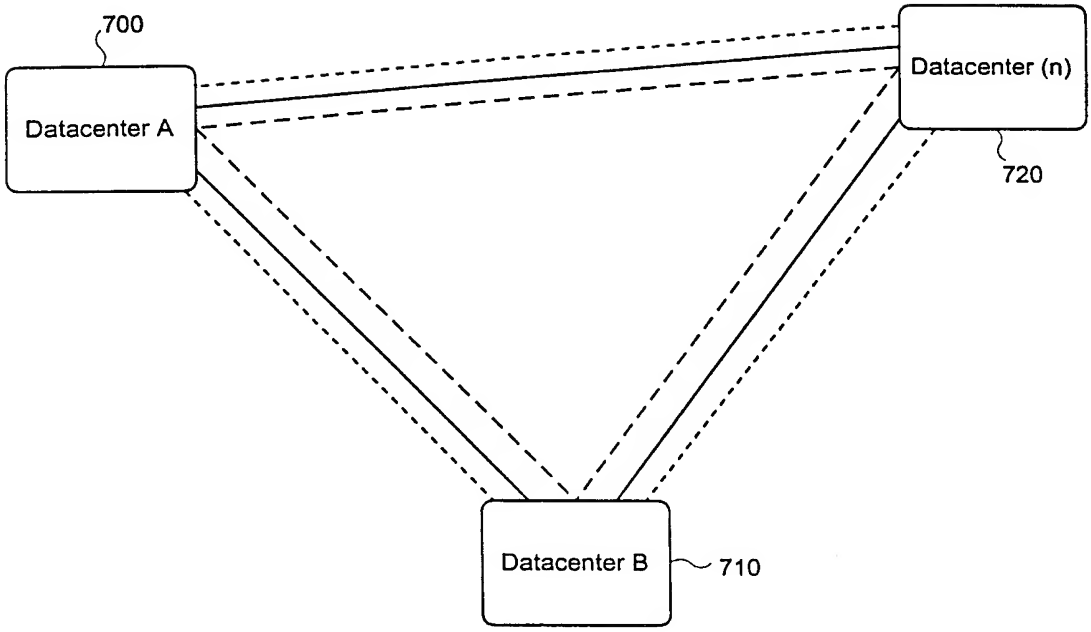


Figure 27

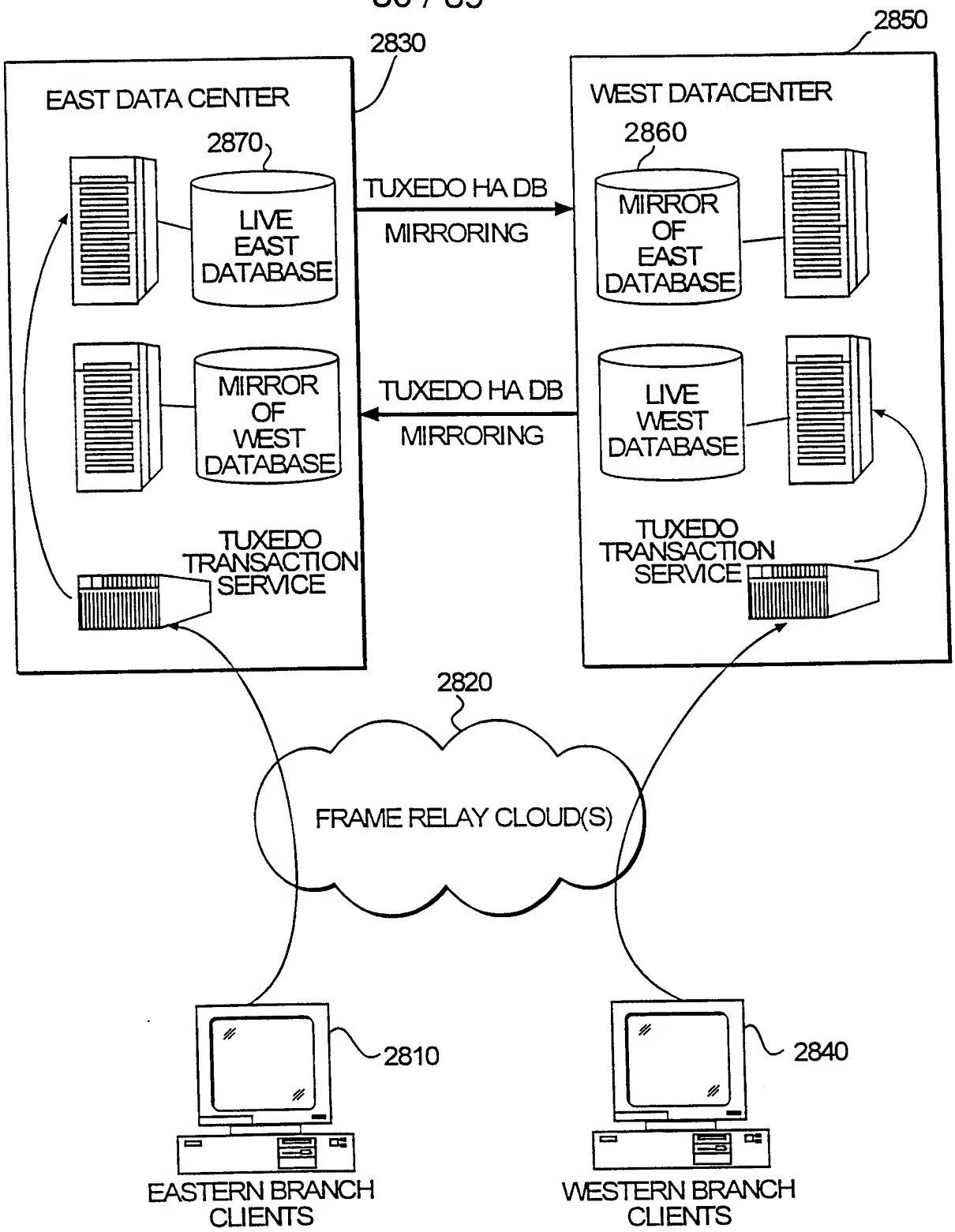

2830
2850
EAST DATA CENTER
WEST DATACENTER
2870
2860
LIVE EAST DATABASE
TUXEDO HA DB MIRRORING
MIRROR OF EAST DATABASE
MIRROR OF WEST DATABASE
TUXEDO HA DB MIRRORING
LIVE WEST DATABASE
TUXEDO TRANSACTION SERVICE
TUXEDO TRANSACTION SERVICE
2820
FRAME RELAY CLOUD(S)
2810
2840
EASTERN BRANCH CLIENTS
WESTERN BRANCH CLIENTS

FIG. 28

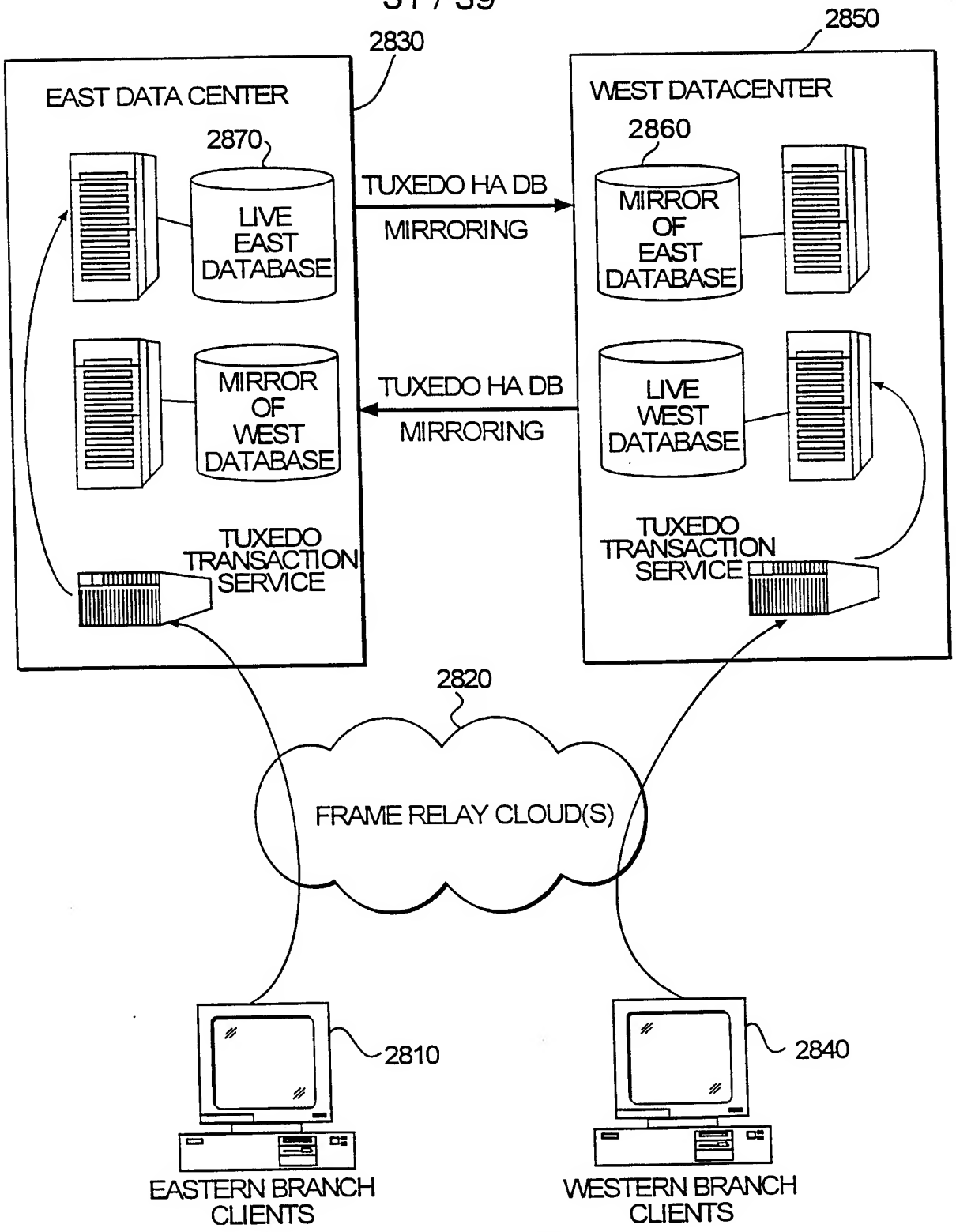

2830
2850
EAST DATA CENTER
WEST DATACENTER
2870
2860
LIVE EAST DATABASE
TUXEDO HA DB MIRRORING
MIRROR OF EAST DATABASE
MIRROR OF WEST DATABASE
TUXEDO HA DB MIRRORING
LIVE WEST DATABASE
TUXEDO TRANSACTION SERVICE
TUXEDO TRANSACTION SERVICE
2820
FRAME RELAY CLOUD(S)
2810
2840
EASTERN BRANCH CLIENTS
WESTERN BRANCH CLIENTS

FIG. 29

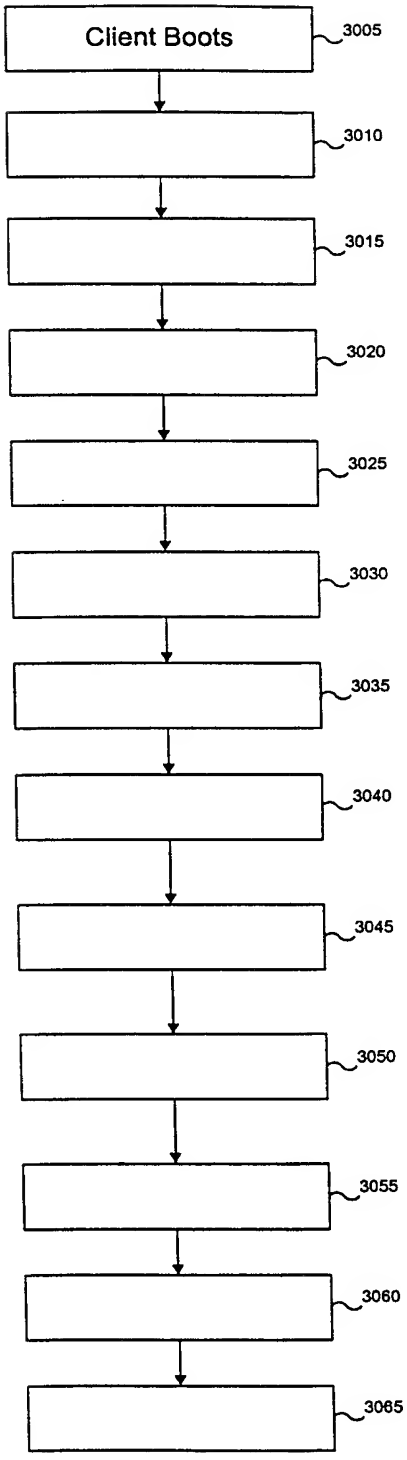

Client Boots
3005
3010
3015
3020
3025
3030
3035
3040
3045
3050
3055
3060
3065

Figure 30

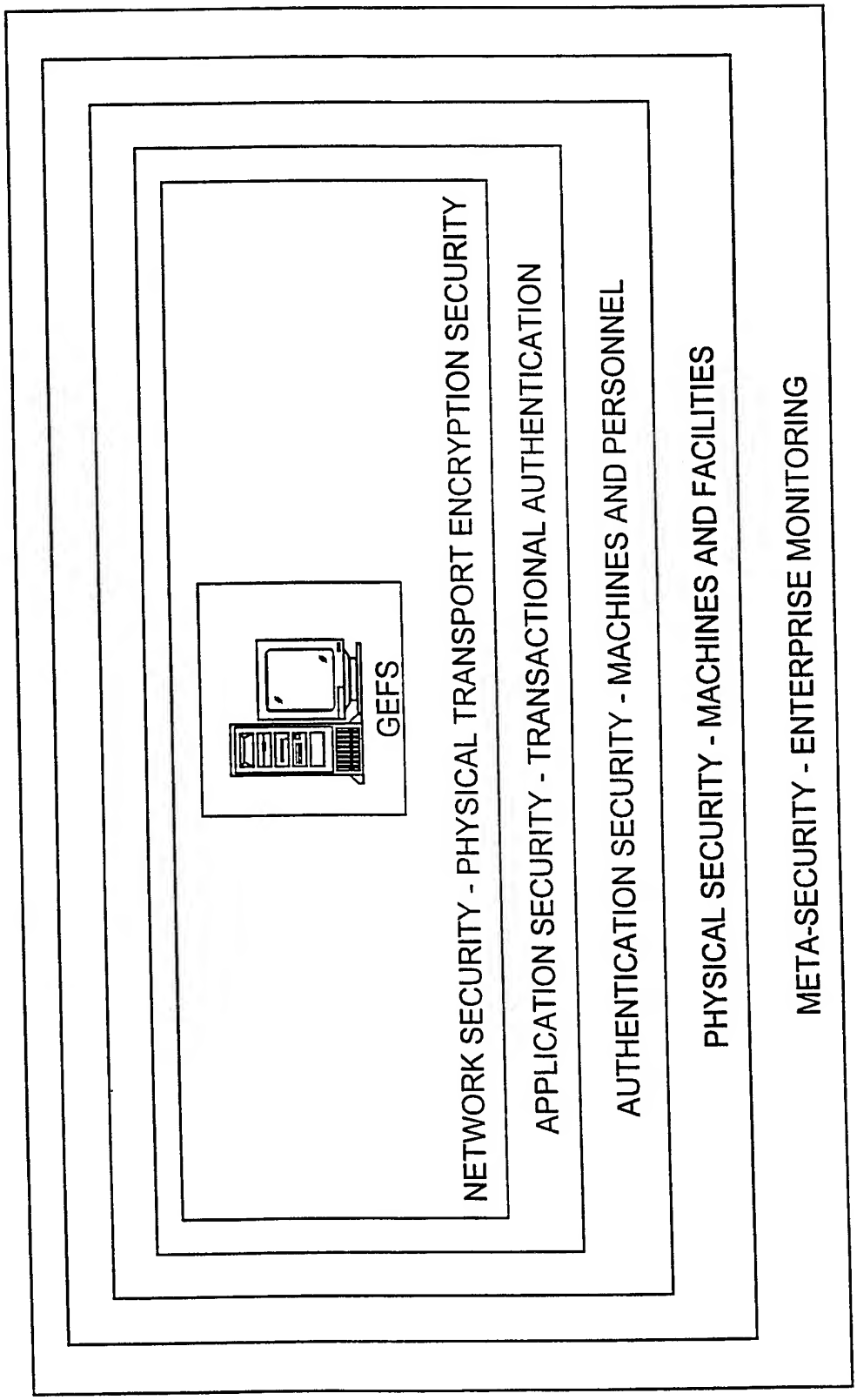

META-SECURITY - ENTERPRISE MONITORING
PHYSICAL SECURITY - MACHINES AND FACILITIES
AUTHENTICATION SECURITY - MACHINES AND PERSONNEL
APPLICATION SECURITY - TRANSACTIONAL AUTHENTICATION
NETWORK SECURITY - PHYSICAL TRANSPORT ENCRYPTION SECURITY
GEFS

FIG. 31

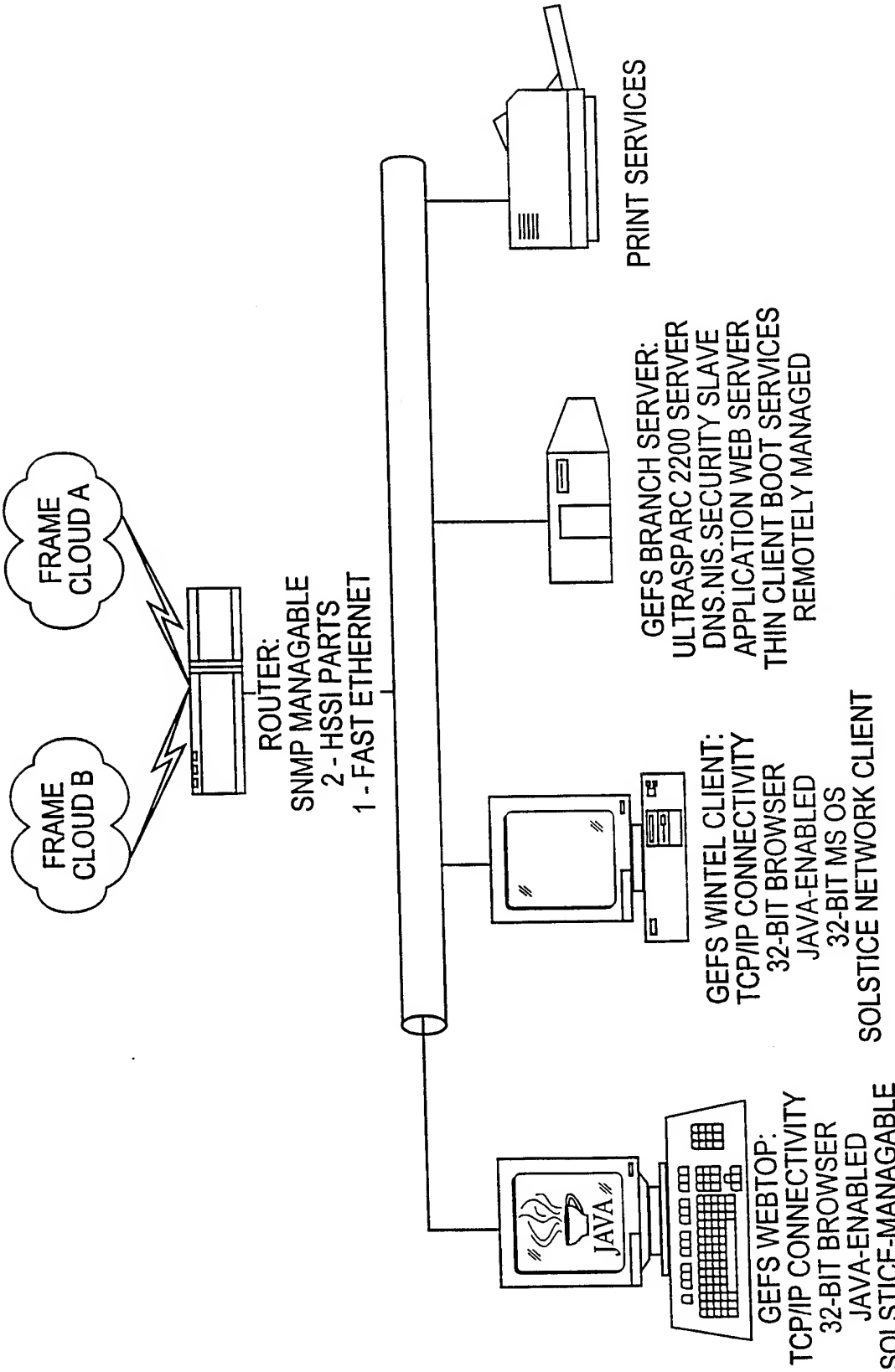

FRAME CLOUD A
FRAME CLOUD B
ROUTER:
SNMP MANAGABLE
2 - HSSI PARTS
1 - FAST ETHERNET
PRINT SERVICES
GEFS BRANCH SERVER:
ULTRASPARC 2200 SERVER
DNS.NIS.SECURITY SLAVE
APPLICATION WEB SERVER
THIN CLIENT BOOT SERVICES
REMOTELY MANAGED
GEFS WINTEL CLIENT:
TCP/IP CONNECTIVITY
32-BIT BROWSER
JAVA-ENABLED
32-BIT MS OS
SOLSTICE NETWORK CLIENT
JAVA
GEFS WEBTOP:
TCP/IP CONNECTIVITY
32-BIT BROWSER
JAVA-ENABLED
SOLSTICE-MANAGABLE
DNS.NIS.BOOTP CLIENT

FIG. 32

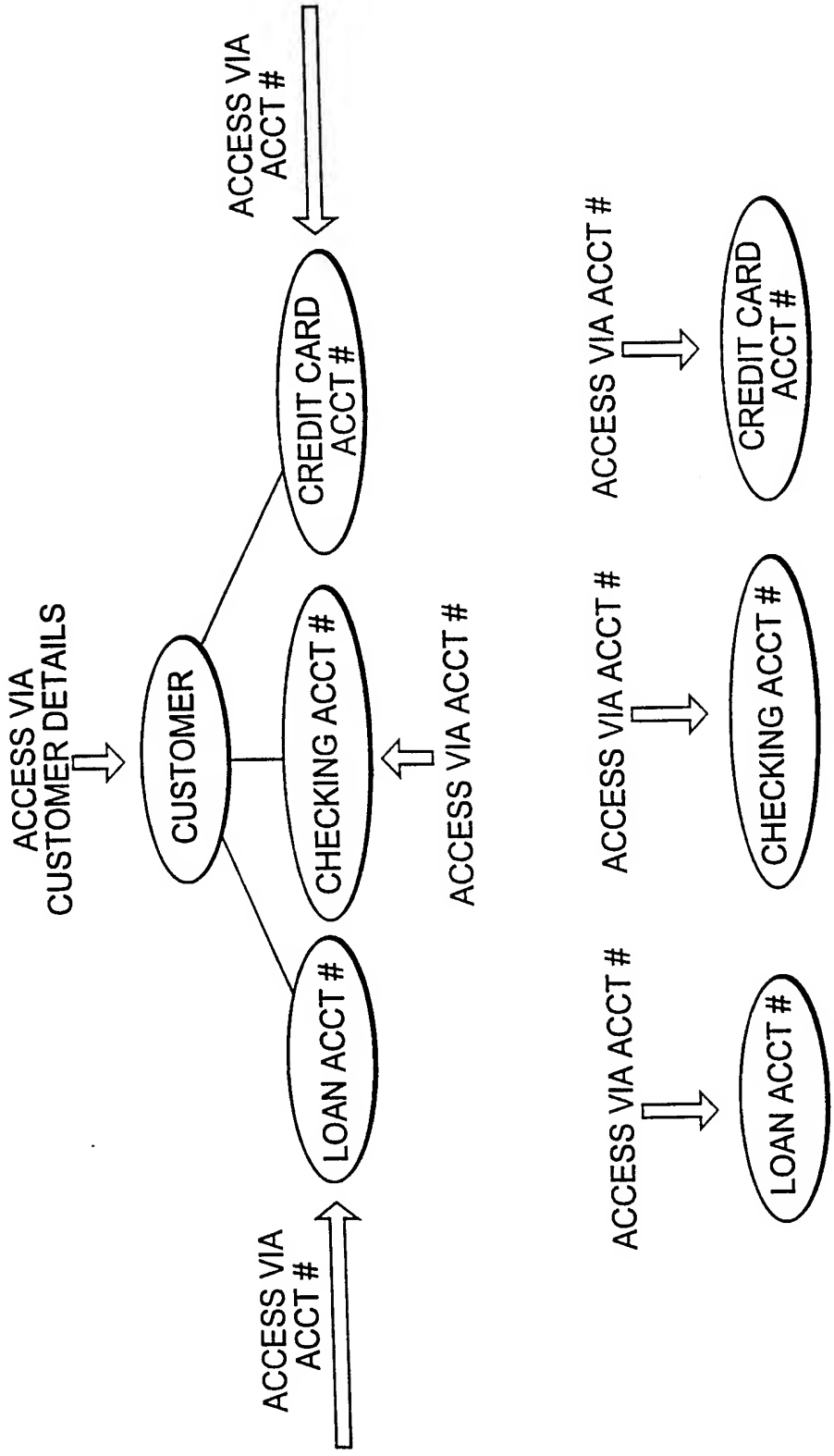

ACCESS VIA CUSTOMER DETAILS
CUSTOMER
ACCESS VIA ACCT #
CREDIT CARD ACCT #
CHECKING ACCT #
ACCESS VIA ACCT #
ACCESS VIA ACCT #
LOAN ACCT #
ACCESS VIA ACCT #
ACCESS VIA ACCT #
CREDIT CARD ACCT #
ACCESS VIA ACCT #
CHECKING ACCT #
ACCESS VIA ACCT #
LOAN ACCT #

FIG. 33

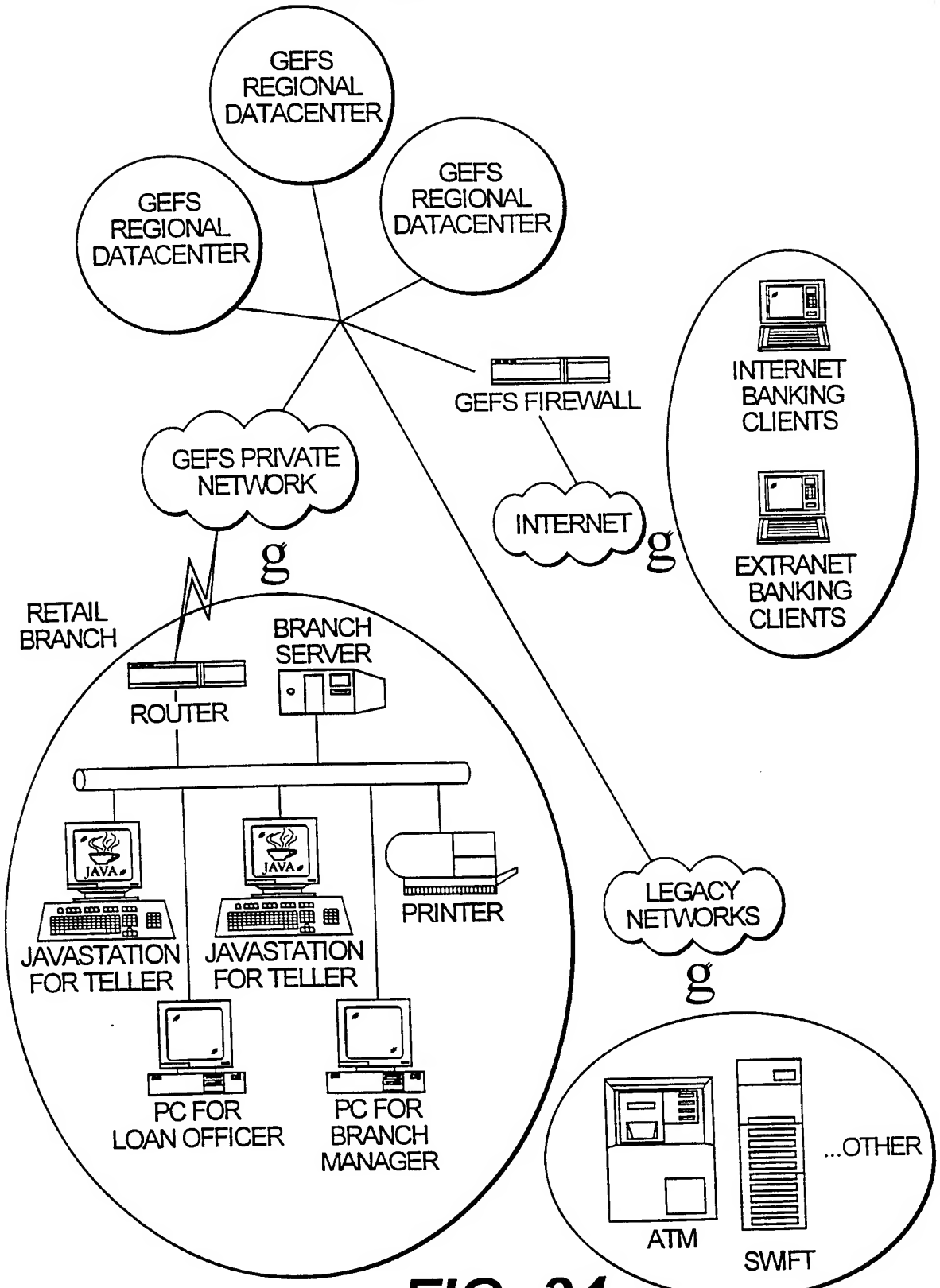

GEFS REGIONAL DATACENTER
GEFS REGIONAL DATACENTER
GEFS REGIONAL DATACENTER
GEFS FIREWALL
INTERNET BANKING CLIENTS
EXTRANET BANKING CLIENTS
GEFS PRIVATE NETWORK
INTERNET
RETAIL BRANCH
BRANCH SERVER
ROUTER
JAVA
JAVA
PRINTER
JAVASTATION FOR TELLER
JAVASTATION FOR TELLER
PC FOR LOAN OFFICER
PC FOR BRANCH MANAGER
LEGACY NETWORKS
ATM
SWIFT
...OTHER

FIG. 34

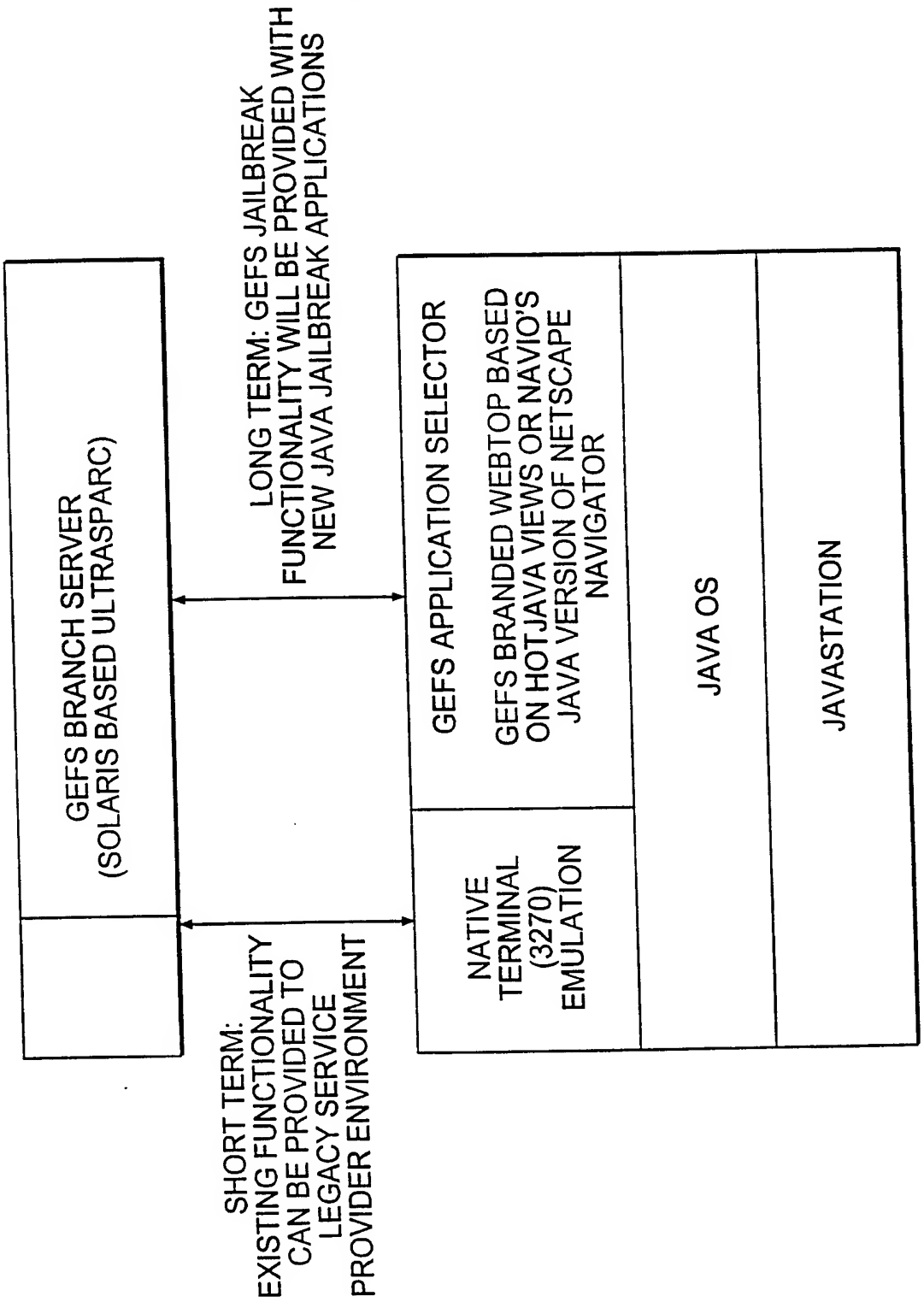

GEFS BRANCH SERVER
(SOLARIS BASED ULTRASPARC)
LONG TERM: GEFS JAILBREAK
FUNCTIONALITY WILL BE PROVIDED WITH
NEW JAVA JAILBREAK APPLICATIONS
GEFS APPLICATION SELECTOR
GEFS BRANDED WEBTOP BASED
ON HOTJAVA VIEWS OR NAVIO'S
JAVA VERSION OF NETSCAPE
NAVIGATOR
JAVA OS
JAVASTATION
NATIVE
TERMINAL
(3270)
EMULATION
SHORT TERM:
EXISTING FUNCTIONALITY
CAN BE PROVIDED TO
LEGACY SERVICE
PROVIDER ENVIRONMENT

FIG. 35

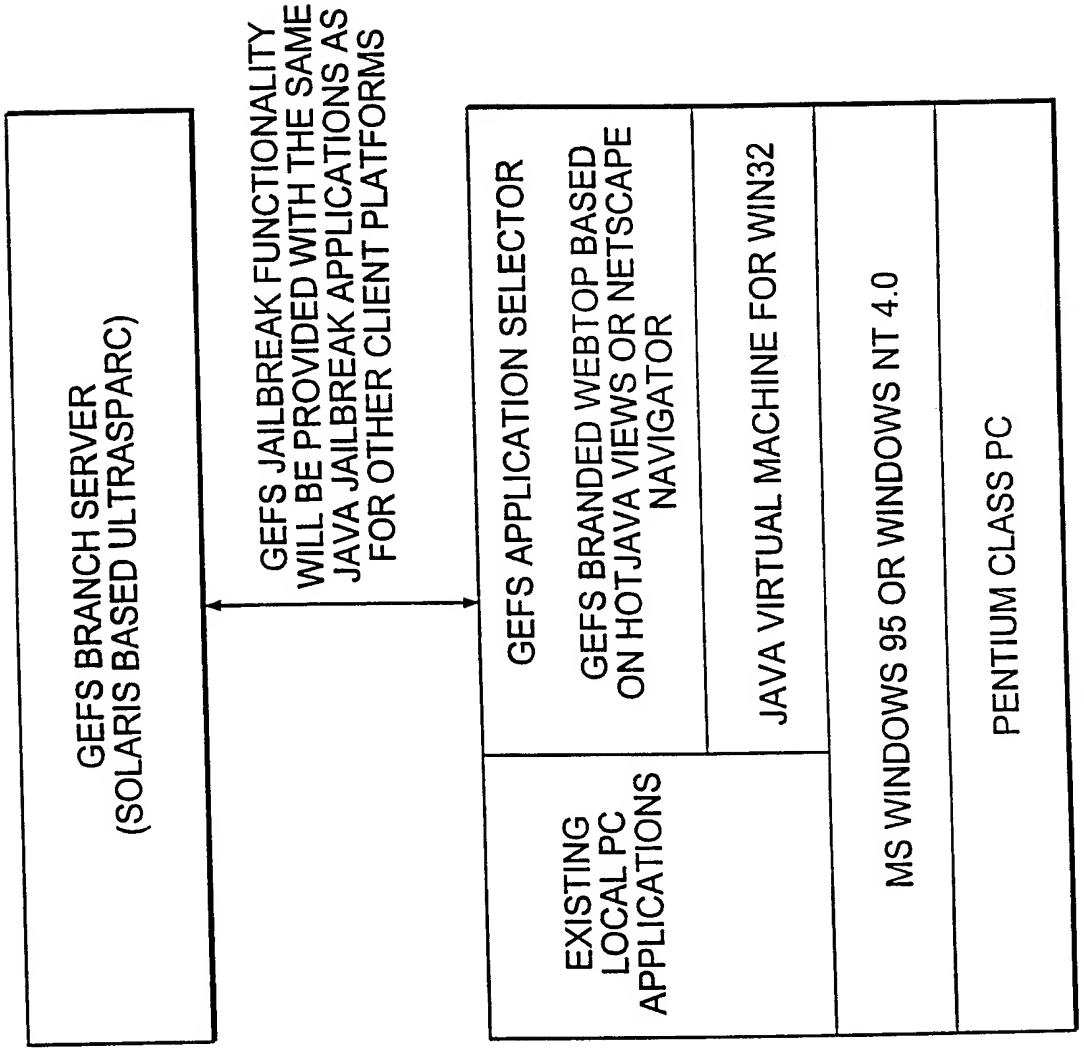

GEFS BRANCH SERVER
(SOLARIS BASED ULTRASPARC)
GEFS JAILBREAK FUNCTIONALITY
WILL BE PROVIDED WITH THE SAME
JAVA JAILBREAK APPLICATIONS AS
FOR OTHER CLIENT PLATFORMS
GEFS APPLICATION SELECTOR
GEFS BRANDED WEBTOP BASED
ON HOTJAVA VIEWS OR NETSCAPE
NAVIGATOR
JAVA VIRTUAL MACHINE FOR WIN32
EXISTING
LOCAL PC
APPLICATIONS
MS WINDOWS 95 OR WINDOWS NT 4.0
PENTIUM CLASS PC

FIG. 36

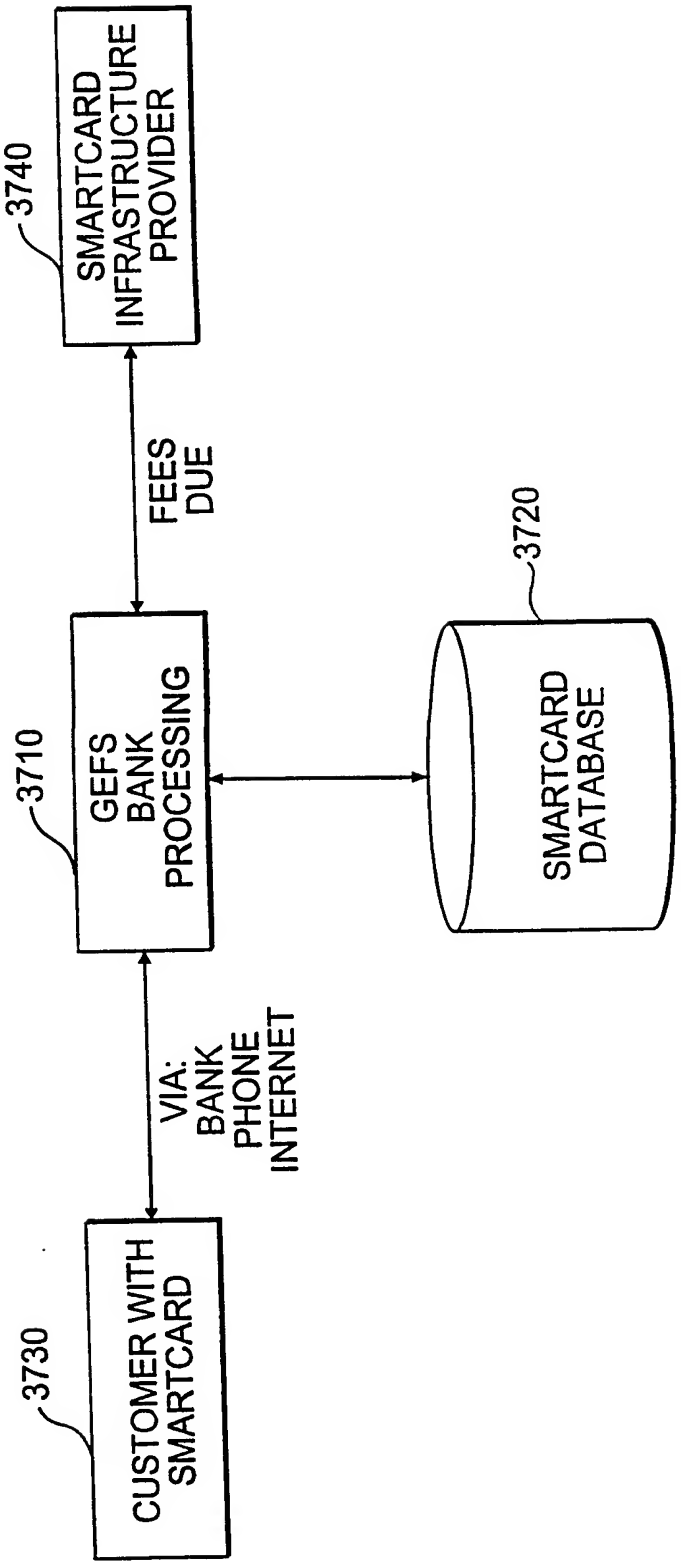

3740
SMARTCARD INFRASTRUCTURE PROVIDER
FEES DUE
3720
3710
GEFS BANK PROCESSING
SMARTCARD DATABASE
VIA: BANK PHONE INTERNET
3730
CUSTOMER WITH SMARTCARD

FIG. 37